68' MICRO JOURNAL

Australia	A	$6.50	New Zealand	NZ	$9.85
Singapore	S	$14.95	Hong Kong	H	$29.00
Malaysia	M	$14.25	Sweden		30:-SEK

$2.95USA

OS-9 Atari Amiga Mac S-50

6800 6809 68008 68000 68010 68020 68030

The Magazine for Motorola CPU Devices *For Over a Decade!*

This Issue:

The Bit Bucket - Motorola News Releases, Letters, Updates etc.

OS-9 SK*DOS Atari Amiga FLEX Macintosh

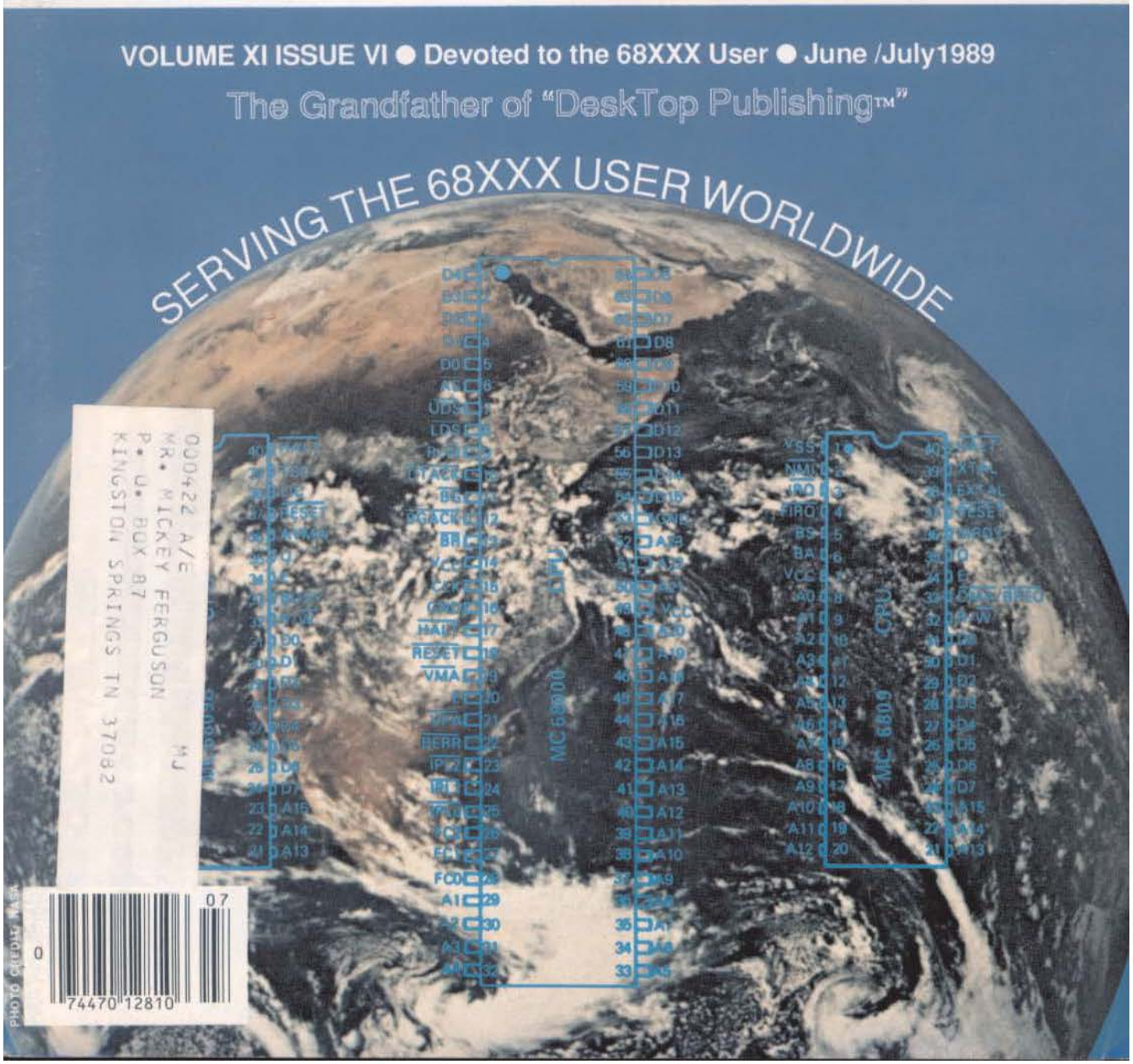
A User Contributor Journal

And Lots More!

VOLUME XI ISSUE VI ● Devoted to the 68XXX User ● June /July1989

The Grandfather of "DeskTop Publishing™"

SERVING THE 68XXX USER WORLDWIDE



000422 A/E
MR. MICKEY FERGUSON MJ
P. O. BOX 87
KINGSTON SPRINGS TN 37082

0 74470 12810 07

PHOTO CREDIT: NASA

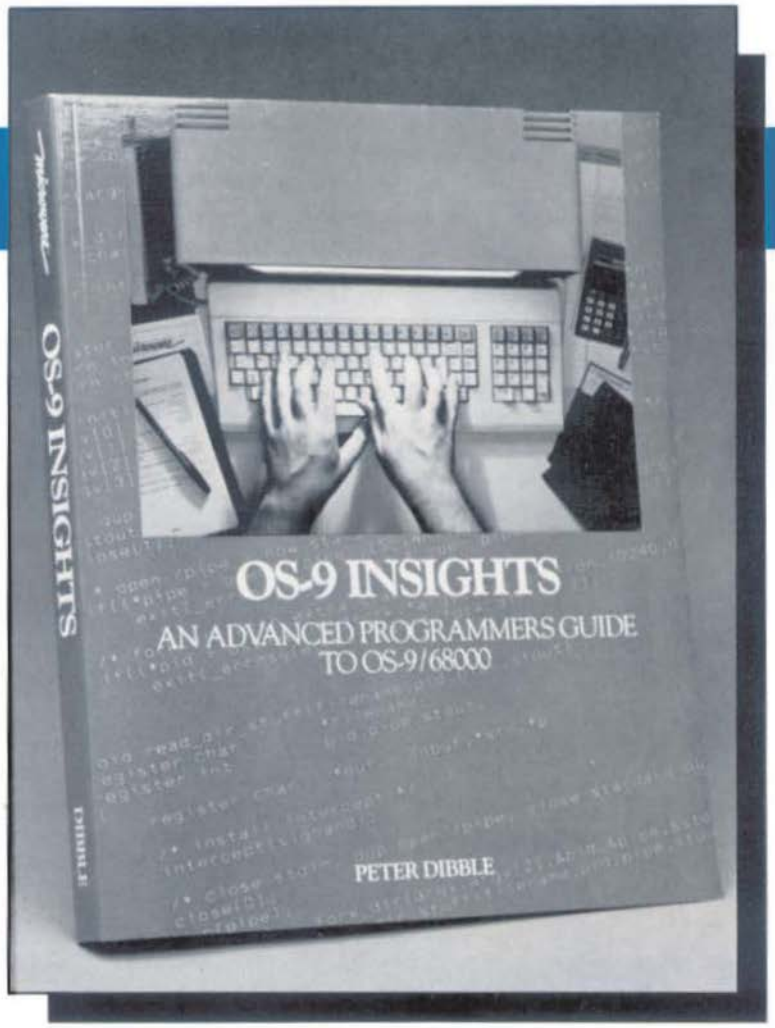
AVAILABLE NOW FROM MICROWARE!

OS-9 INSIGHTS

An Advanced Programmers Guide To OS-9/68000

- **An in-depth examination of the OS-9 design philosophy**
- **Detailed discussion of Kernel operation and real-time features**
- **Sample file manager and driver source listings**
- **Information on customizing your OS-9 system**
- **An invaluable tutorial for the professional programmer**

"This book was written for programmers who would like to use the advanced features of OS-9. It explains and illustrates features of OS-9 ranging from memory management through file managers."
—Peter Dibble



This is a "must" book for all serious OS-9 programmers!

Use this handy order form

Name _____
Address _____
City _____
State _____ Zip _____

Please send me _____ copy(s) of OS-9 Insights at $40.00 each. Add $2.00 shipping and handling for each copy ordered.

TOTAL (includes shipping) $ _____

Fill in all the information on this order form. Mail the completed order form along with your check, money order (no cash, please) or credit card information to:

microware®

MICROWARE SYSTEMS CORPORATION
1900 N.W. 114th Street • Des Moines, Iowa 50322
Attention: Order Department
Or Call: 515-224-1929

Credit card customers must complete the following:
(Check One)
☐ MasterCard ☐ VISA
Credit Card # ____ ____ ____ ____
Expiration Date ____/____
Signature _____
(Required — Credit Card Customers Only)

DON'T DELAY — ORDER TODAY!

Mustang-020 Mustang-08 Benchmarks

	32 bit Integer	Register Long
IBM AT 7300 Xenix Sys 3	9.7	
AT&T 7300 UNIX PC 68010	7.2	4.3
DEC VAX 11/780 UNIX Berkley 4.2	3.6	3.2
DEC VAX 11/750	3.1	3.2
68008 OS-9 68K 8 Mhz	18.0	9.0
68000 OS-9 68K 10 Mhz	6.5	4.0
MUSTANG-08 68008 OS-9 68K 10 Mhz	***9.8***	***6.3***
MUSTANG-020 68020 OS-9 68K 16 Mhz	***2.2***	***0.88***
MUSTANG-020 68020 MC68881 UniFLEX 16 Mhz	***1.8***	***1.22***

```
Main()
        {
            register long i;
            for (i=0; i < 999999; ++i);
        }
```

Estimated MIPS - MUSTANG-020 — 4.5 MIPS,

Burst to 8 - 10 MIPS: Motorola Specs

Mustang Specifications

12.5 Mhz (optional 16.6 Mhz available) MC68020 full 32-bit wide path
32-bit wide data and address buses, non-multiplexed
on chip instruction cache
object code compatible with all 68XXX family processors
enhanced instruction set - math co-processor interface
68881 math hi-speed floating point co-processor (optional)
direct extension of full 68020 instruction set
full support IEEE P754, draft 10.0
transcendental and other scientific math functions
2 Megabyte of SIP RAM (512 x 32 bit organization)
up to 256K bytes of EPROM (64 x 32 bits)
4 Asynchronous serial I/O ports standard
optional to 20 serial ports
standard RS-232 interface
optional network interface
buffered 8 bit parallel port (1/2 MC68230)
Centronics type pinout
expansion connector for I/O devices
16 bit data path
256 byte address space
2 interrupt inputs
clock and control signals
Motorola I/O Channel Modules
time of day clock/calendar w/battery backup
controller for 2, 5 1/4" floppy disk drives
single or double side, single or double density
35 to 80 track selectable (48-96 TPI)
SASI interface
programmable periodic interrupt generator
interrupt rate from micro-seconds to seconds
highly accurate time base (5 PPM)
5 bit sense switch, readable by the CPU
Hardware single-step capability

CPI
Mustang Hi-Speed Systems
Only from Data-Comp Div.
68008-68030

Don't be mislead!
ONLY Data-Comp delivers the Super MUSTANG-020

OS-9

OS-9 Professional Ver *includes C Compiler	$850.00
Basic09	450.00
C Compiler	500.00
68000 Disassembler (w/source add: $100.00)	100.00
Fortran 77	750.00
Microware Pascal	500.00
Omegasoft Pascal	900.00
Stylo-Graph	495.00
Stylo-Spell	195.00
Stylo-Merge	175.00
Stylo-Graph.Spell.Merge	695.00
PAT w/C source	229.00
JUST w/C source	79.95
PAT/JUST Combo	249.50
Sculptor+ (see below)	995.00
COM	125.00

UniFLEX

UniFLEX (68020 ver)	$450.00
Screen Editor	150.00
Sort-Merge	200.00
BASIC/PreCompiler	300.00
C Compiler	350.00
COBOL	750.00
CMODEM w/source	100.00
TMODEM w/source	100.00
X-TALK (ver Ad)	99.95
Cross Assembler	50.00
Fortran 77	450.00
Sculptor+ (see below)	995.00

Standard MUSTANG-020™ shipped 12.5 Mhz.

Add for 16.6 Mhz 68020	375.00
Add for 16.6 Mhz 68881	375.00
Add for 20 Mhz 68020/RAM	750.00
16 Port exp. RS-232	335.00
Requires 1 or 2 Adapter Cards below RS232 Adapter	165.00

Each card supports 4 additional ser. ports (total of 36 serial ports supported)

60 line Parallel I/O card	398.00

Uses 3 68230 Interface/Timer chips, 6 groups of 8 lines each, separate buffer direction control for each group.

Prototype Board	75.00

areas for both dip and PGA devices & a pre-wired memory area up to 512K DRAM.

SBC-AN	475.00

Interface between the system and ARCNET modified token-passing LAN, fiber optics optional - call. LAN software drivers 120.00

Expansion for Motorola I/O Channel Modules	$195.00

special for complete MUSTANG-020™ system buyers - Sculptor+ $695.00. SAVE $300.00

Software Discounts

All MUSTANG-020™ system and board buyers are entitled to discounts on all listed software: 10-70% depending on item. Call or write for quotes. Discounts apply after the sale as well.

The PRO!

These hi-speed 68020 systems are presently working at NASA, Atomic Energy Commission, Government Agencies as well as Universities, Business, Labs, and other Critical Applications Centers, worldwide, where speed, math crunching and multi-user, multi-tasking UNIX C level V compatability and low cost is a must.

Only the "PRO" Version of OS-9 Supported!

This is *HEAVY DUTY* Country!

For a limited time we will offer a $400 trade-in on your old 68XXX SBC. Must be working properly and complete with all software, cables and documentation. Call for more information

Price List:

Mustang-020 SBC		$2490.00
Cabinet w/switching PS		$299.95
5"-80 track floppy DS/DD		$269.95
Floppy Cable		$39.95
OS-9 68K Professional Version		$850.00
C Compiler ($500 Value)		N/C
Winchester Cable		$39.95
Winchester Drive 25 Mbyte		$895.00
Hard Disk Controller		$395.00
Shipping USA UPS		$20.00
UniFLEX	Less	$100.00
MC68881 f/p math processor	Add	$275.00
16.67 Mhz MC68020		$375.00
16.67 Mhz MC68881		$375.00
20 Mhz MC68020 Sys		$750.00
Note all 68881 chips work with 20 Mhz Sys		
	Total:	$5299.80

NEW LOWER PRICES

25 Mbyte HD ~~$4299.80~~ $3749.80

85 Mbyte HD ~~$5248.80~~ $4548.80

Data-Comp Division

"A Decade of Quality Service"
Systems World-Wide

Computer Publishing, Inc. 5900 Cassandra Smith Road
Hixson, Tn 37343
Telephone (615) 842-4601 - FAX (615) 842-7990

Note: Only Professional OS-9 Now Available (68020 Version) Includes ($500) C Compiler - 68020 & 68881 Supported - For UPGRADES Write or Call for Professional OS-9 Upgrade Kit

A Member of the CPI Family

68 Micro Journal

10 Years of Dedication to Motorola CPU Users

6800 6809 68000 68010 68020

The Originator of "DeskTop Publishing™"

Publisher
Don Williams Sr.

Executive Editor
Larry Williams

Production Manager
Tom Williams

Office Manager
Joyce Williams

Subscriptions
Cheryl Hodge

Contributing & Associate Editors

Ron Anderson
Ron Voigts
Doug Lurie
Ed Law
Dr. E.M. "Bud" Pass
Art Weller
Dr. Theo Elbert
& Hundreds More of Us

Contents

68 MICRO JOURNAL

"Contribute Nothing - Expect Nothing" DMW 1986

COMPUTER PUBLISHING, INC.

"Over a Decade of Service"

"World Wide"

68 MICRO JOURNAL
Computer Publishing Center
5900 Cassandra Smith Road
PO Box 849
Hixson, TN 37343

Phone (615) 842-4600 FAX (615) 842-7990

Copyrighted © 1987 by Computer Publishing, Inc.

68 Micro Journal is the *original* "DeskTop Publishing" product and has continously published since 1978 using only micro-computers and special "DeskTop" software. Using first a kit built 6800 micro-computer, a modified "ball" typewriter, and "home grown" DeskTop Publishing software. None was commercially available at that time. For over 10 years we have been doing "DeskTop Publishing"! *We originated what has become traditional "DeskTop Publishing"!* Today 68 Micro Journal is acknowledged as the "Grandfather" of "DeskTop Publishing" technology.

---

68 Micro Journal (ISSN 0194-5025) is published 12 times a year by Computer Publishing Inc. Second Class Postage paid at Hixson, TN. and additional entries. POSTMASTER: send address changes to 68 Micro Journal, POB 849, Hixson, TN 37343.

Subscription Rates

1 Year $24.50 USA, Canada & Mexico $34.00 a year. Others add $12.00 a year surface, $48.00 a year Airmail, USA funds. 2 years $42.50, 3 years $64.50 plus additional postage for each additional year.

Items or Articles for Publication

Articles submitted for publication must include authors name, address, telephone number, date and a statement that the material is original and the property of the author. Articles submitted should be on diskette, OS-9, SK*DOS, FLEX, Macintosh or MS-DOS. All printed items should be dark type and satisfactory for photo-reproduction. No blue ink! No hand written articles - please! Diagrams o.k.

Please - do not format with spaces any text indents, charts, etc. (source listing o.k.). We will edit in all formatting. Text should all flush left and use a carriage return only to indicate a paragraph end. Please write for free authors guide.

Letters & Advertising Copy

Letters to the Editor should be the original copy, signed! Letters of grip as well as praise are acceptable. *We reserve the right to reject any letter or advertising material, for any reason we deem advisable.* Advertising Rates: Commercial please contact 68 Micro Journal Advertising Department. Classified advertising must be non-commercial. Minimum of $15.50 for first 15 words. Add $.60 per word thereafter. No classifieds accepted by telephone.

PAT - JUST

PAT WITH 'C' Source $229.00
JUST WITH 'C' Source $79.95

OS-9 68K - 68008 - 68000 - 68010 - 68020 - OS-9 68K

PAT FROM S. E. MEDIA — A FULL FEATURED SCREEN ORIENTED TEXT EDITOR with all the best of PIE. For those who swore by and loved PIE, this is for **YOU! All PIE features & much more!** Too many features to list. And if you don't like ours, change or add your own. C source included. Easily configures to your CRT terminal, with special configuration section. No sweat!

COMBO
PAT
JUST
Special $249.00

JUST from S. E. MEDIA — Text formatter written by Ron Anderson; for dot matrix printers, provides many unique features. Output formatted to the display. User configurable for adapting to other printers. Comes set -up for Epson MX80 with Graflex. Up to 10 imbedded printer control commands. Compensates for double width printing. Includes normal line width, page numbering, margin, indent, paragraph, space, vertical skip lines, page length, centering, fill, justification, etc. Use with **PAT** or any other text editor. The ONLY stand alone text processor for the 68XXX OS-9, that we have seen. And at a very **LOW PRICE!** Order from: S. E. MEDIA - see catalog this issue.

MASTER CARD

VISA

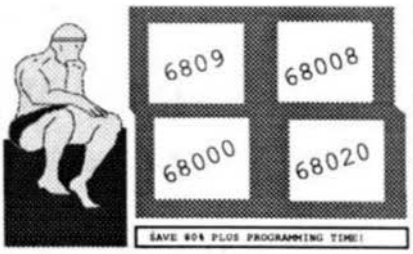
Southeast East Media
5900 Cassandra Smith Rd
Hixson, Tn. 37343
Telephone (615) 842-4600
FAX (615) 842-7990

Shipping

U.S.A.	$4.50
CANADA	$7.50
FOREIGN	$25.00

CLOSE OUT SPECIAL

SCULPTOR



From the world's oldest & largest OS-9 software house!

CUTS PROGRAMMING TIME UP TO 80%

6809/68000-68030 Save 90%

SCULPTOR-a 4GL - Only from S.E. Media at these prices. OS-9 levels one and two (three GIMIX) 6809, all 68XXX OS-9 standard systems. Regular SCULPTOR versions 1.4:6. One of if not the most efficient and easy to develope DBMS type systems running under OS-9. A system of flexible keyed file access that allows extremely fast record and data retrieval, insertion and deletion or other programmed modifications. Access by key or in ascending order, very fast. The system provides automatic menu generation, compilation and report generation. Practically unlimited custom input format and report formatting. A rich set of maintenance and repair utilities. An extremely efficient development environment that cuts most programming approximately 80% in development and debugging! Portable, at source level, to MS-DOS, UNIX and many other languages and systems.

Standard Version: 1.14:6

6809 - $1295.00
68000 $1295.00
68020 $1990.00

Due to a "Special One Time" Purchase, We Are Making This Savings Offer. Quantities Limited!

Once this supply is gone the price goes back up!

System OS-9: 6809/68000-68030

• Regular ~~$1295.00~~

ONLY **$99.95**

S.E. MEDIA

POB 849 5900 CASSANDRA SMITH ROAD
HIXSON, TN 37343
TELEPHONE(615) 842-4601 FAX (615)842-7990

$AVE - WHILE SUPPLIES LAST!

Telephone: (615) 842-4600

South East Media

OS-9, UniFLEX, FLEX, SK*DOS
SOFTWARE

Fax: (615) 842-7990

SCULPTOR

Full OEM & Dealer Discounts Available!

THE SCULPTOR SYSTEM

Sculptor combines a powerful fourth generation language with an efficient database management system. Programmers currently using traditional languages such as Basic and Cobol will be amazed at what Sculptor does to their productivity. With Sculptor you'll find that what used to take a week can be achieved in just a few hours.

AN ESTABLISHED LEADER

Sculptor was developed by professionals who needed a software development tool with capabilities that were not available in the software market. It was launched in 1981 and since then, with feedback from an ever-increasing customer base, Sculptor has been refined and enhanced to become one of the most adaptable, fast, and above all reliable systems on the market today.

SYSTEM INDEPENDENCE

Sculptor is available on many different machines and for most operating systems, including MS DOS, Unix/Xenix and VMS. The extensive list of supported hardware ranges from small personal computers, through multi-user micros up to large minis and mainframes. Sculptor is constantly being ported to new systems.

APPLICATION PORTABILITY

Mobility of software between different environments is one of Sculptor's major advantages. You can develop applications on a stand-alone PC and – without any alterations to the programs – run them on a large multi-user system. For software writers this means that their products can reach a wider marketplace than ever before. It is this system portability, together with high-speed development, that makes Sculptor so appealing to value added resellers, hardware manufacturers and software developers of all kinds.

SPEED AND EFFICIENCY

Sculptor uses a fast and proven indexing technique which provides instant retrieval of data from even the largest of files. Sculptor's fourth generation language is compiled to a compact intermediate code which executes with impressive speed.

INTERNATIONALLY ACCEPTED

By using a simple configuration utility, Sculptor can present information in the language and format that you require. This makes it an ideal product for software development almost anywhere in the world. Australasia, the Americas and Europe – Sculptor is already at work in over 20 countries.

THE PACKAGE

With every development system you receive:
- ☐ A manual that makes sense
- ☐ A periodic newsletter
- ☐ Screen form language
- ☐ Report generator
- ☐ Menu system
- ☐ Query facility
- ☐ Set of utility programs
- ☐ Sample programs

For resale products, the run-time system is available at a nominal cost.

Facts **Features**

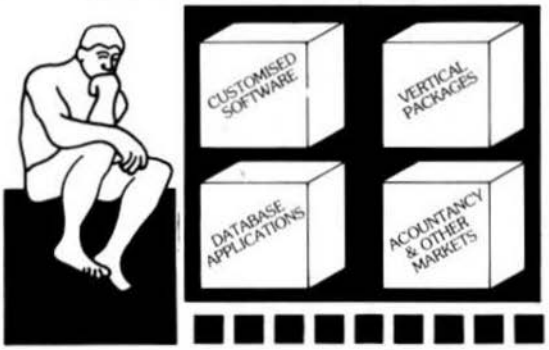
DATA DICTIONARY

Each file may have one or more record types described. Fields may have a name, heading, type, size, format and validation list. Field type may be chosen from:
- ☐ alphanumeric
- ☐ integer
- ☐ floating point
- ☐ money
- ☐ date

DATA FILE STRUCTURE

- ☐ Packed, fixed-length records
- ☐ Money stored in lower currency unit
- ☐ Dates stored as integer day numbers

Sculptor for 68020 OS-9 & UniFLEX $995



INDEXING TECHNIQUE

Sculptor maintains a B-tree index for each data file. Program logic allows any numbers of alternative indexes to be coded into one other file.

INPUT DATA VALIDATION

Input data may be validated at three levels:
- ☐ automatic by field type
- ☐ validation list in data dictionary
- ☐ programmer coded logic

ARITHMETIC OPERATORS

–	Unary minus
*	Multiplication
/	Division
%	Remainder
+	Addition
–	Subtraction

MAXIMA AND MINIMA

Minimum key length	1 byte
Maximum key length	160 bytes
Minimum record length	3 bytes
Maximum record length	32767 bytes
Maximum fields per record	32767
Maximum records per file	16 million
Maximum files per program	16
Maximum open files	Operating system limit

PROGRAMS

- ☐ Define record layout
- ☐ Create new indexed file
- ☐ Generate standard screen-form program
- ☐ Generate standard report program
- ☐ Compile screen-form program
- ☐ Compile report program
- ☐ Screen-form program interpreter
- ☐ Report program interpreter
- ☐ Menu interpreter

RELATIONAL OPERATORS

=	Equal to
<	Less than
>	Greater than
< =	Less than or equal to
> =	Greater than or equal to
< >	Not equal to
and	Logical and
or	Logical or
ct	Contains
bw	Begins with

SPECIAL FEATURES

- ☐ Full date arithmetic
- ☐ Echo suppression for passwords
- ☐ Terminal and printer independence
- ☐ Parameter passing to sub-programs
- ☐ User definable date format
- ☐ Query facility
- ☐ Reformat file
- ☐ Check file integrity
- ☐ Rebuild index
- ☐ Alter language and date format
- ☐ Setup terminal characteristics
- ☐ Setup printer characteristics

SCREEN-FORM LANGUAGE

- ☐ Programmer defined options and logic
- ☐ Multiple files open in one program
- ☐ Default or programmer processing of exception conditions
- ☐ Powerful verbs for input, display and file access
- ☐ Simultaneous display of multiple records
- ☐ Facility to call sub-programs and operating system commands
- ☐ Conditional statements
- ☐ Subroutines
- ☐ Independent of terminal type

MUSTANG-020 Users - Ask For Your Special Discount!

	*	**	***		*	**	***
MUSTANG-020	***$1,990**	**$398**	**$795**	**PC/XT/AT/MSDOS**	**$695**	**$139**	**$299**
MUSTANG-08	***$1,295**	**$259**	**$495**	**Call or write for prices on the following systems.**			

XENIX SYS III & V, MS-NET, UNIX SYS III & V, ATARI OS-9, 68K, UNOS, ULTRIX/VMS (VAX,REGAL), STRIDE, ALTOS, APRICORT, ARETE, ARM-STRONG, BLEASDALE, CHARLES RIVERS, CMX, CONVERG.TECH, DEC, CIFER, EQUINOX, GOULD, HP, HONEYWELL, IBM, INTEL, MEGADATA, MOTOROLA, NCR, NIXDORF, N.STAR, OLIVETTI/AT&T, ICL, PERKINS ELMER, PHILLIPS, PIXEL, PLESSEY, PLEXUS, POSITRON, PRIME, SEQUENT, SIEMENS, SWTPC, SYSTIME, TANDY, TORCH, UNISYS, ZYLOG, ETC.

*** For SPECIAL LOW SCULPTOR prices especially for 6809/68XXX OS-9 Systems - See Special Ad this issue. Remember, "When they are gone the price goes back up as above!"**

... Sculptor Will Run On Over 100 Other Types of Machines ...
... Call for Pricing ...
!!! Please Specify Your Make of Computer and Operating System !!!

* Full Developement Package
** Run Time Only
*** C Key File Library

Availability Legends
O = OS-9, S = SK*DOS
F = FLEX, U = UniFLEX
CO9 = Color Computer OS-9
COF = Color Computer FLEX

VISA

South East Media
5900 Cassandra Smith Rd. - Hixson, Tn. 37343
Telephone: (615) 842-4600 FAX (615) 842-7990

master card

** Shipping **
Add 2% U.S.A. (min. $2.50)
Foreign Surface Add 5%
Foreign Airmail Add 10%
Or C.O.D. Shipping Only

*OS-9 is a Trademark of Microware and Motorola-*FLEX and UniFLEX are Trademarks of Technical Systems Consultants-*SK*DOS is a Trademark of Star-K Software Systems Corp.

Basically OS-9

Dedicated to the serious OS-9 user.
The fastest growing users group world-wide!
6809 - 68020

A Tutorial Series

By: Ron Voigts
2024 Baldwin Court
Glendale Heights, IL 60139

WINDOWS, COLORS AND ALARMS

Last time, I left you in a discussion of OS-9 windows. I had covered some of the basics that should get you started. Basically everything is built on device windows. Device windows are designated as /W1 to /W7. They are used for running programs. In a sense, they give you individual terminals. So your Coco computer can act like 8 terminals running OS-9. The device window and its boundaries cannot overlap another.

An overlay window is one which is created over an existing device window. Overlay windows can overlap. They are primarily used for computer dialogue. I/O going to the device window will channel to the overlay window.

Last time I presented you with 4 C Library routines. The were owset(), owend(), curon() and curoff(). These first two opened and closed overlay windows. The later two turned the cursor on and off.

The easiest way to create a device window is with WCREATE. Usually it is put into the STARTUP file. I have my STARTUP create 3 window. One is 24X40 and two are 24X80. I keep the small screen for general purpose I/O. The other two are for word processing and running compilers. The syntax of WCREATE is:

wcreate /path -s=type xpos ypos xsize ysize foreground background [border]

/path - a device window that is available, ie. /W1 and /W2

xpos - x position of upper left hand corner of window

ypos - y position of upper left hand corner of window

xsize - number of columns for window

ysize - number of lines for window

foreground - window's foreground pallette color

background - window's background pallette color

border - window's border pallette color

type - the type of window to create

There are 6 different types of windows. They range in size, memory requirement and graphic type. The following table summarizes them.

TYPE	SIZE	RESOLUTION	COLORS	MEMORY	SCREEN
1	40X24		16	1600	Text
2	80X24		16	4000	Text
5	80X24	640X192	2	16000	Graphics
6	40X24	320X192	4	16000	Graphics
7	80X24	640X192	4	32000	Graphics
8	80X24	320X192	16	32000	Graphics

An example of my STARTUP file reads:

```
iniz /w1
wcreate /w1 -s=2 0 0 80 24 5 10 10
shell -i=/w1 &
```

INIZ attaches the /W1 descriptor to OS-9. It puts the device into the device table, allocates the necessary memory and calls the device driver to initialize the device. WCREATE then creates the window for /W1. This window is 80X24 characters in size taking up the whole screen. the foreground is amber. The background and border are black. The SHELL creates an immortal shell on /W1.

I usually create a few windows like this. I keep the original screen as a safe guard against hungups. If I should do something to latch up the screen I'm on, I can move back to the original screen and see what I can do. (To move forward a screen type<CLEAR>; to move backward type <SHIFT><CLEAR>.) There are times when the hangup is a total disaster. A good sign of this is interesting display of colors and characters when they weren't

intended.

Speaking of colors, unlike the old Color Computer which was severely limited, the Color Computer 3 has 64 different colors available. 16 palettes are available for colors. It is up to you what you want in them. The default colors at startup are:

REGISTER	COLOR	REGISTER	COLOR
00	Black	08	Black
01	Red	09	Green
02	Green	10	Black
03	Yellow	11	Buff
04	Blue	12	Black
05	Magenta	13	Green
06	Cyan	14	Black
07	White	15	Orange

Do these colors sound familiar? They are the basic colors that came with the original Color Computer. The 00 to 07 colors were the standard graphic colors. 08 to 15 were the alpha/numeric screen colors. Don't remember the Orange/Black screen, eh? There was one. It was a bit tricky to implement, but you could get it.

The colors are determined by the bit set in the byte of the palette location. The colors are:

Bit Color
0 Blue (LSB)
1 Green
2 Red
3 Blue
4 Green
5 Red
6 Not used
7 Not used (MSB)

These combinations will give the 64 colors I mentioned earlier. The way to use these is to send the following byte sequence to the terminal.

$1B $31 [Palette #] [Color Code]

DISPLAY is the easiest way to send these. Let us say we want to change palette position 0 to a bright red enter:

display 1B 31 00 24

Here $24 is the bit pattern %00100100. Try either $04 or $20. The bit color with be some variation of red. Try mixing the colors. Send a $3F. This sets all the bits high and the color is white. What will $00 do?

The color patterns I give here are for an RGB monitor. Composite video will have some variations of the colors. A little experimentation can be fun.

I am going to leave you with a program called ALARM. It basically sets a free running alarm. At the appointed time it beeps and signals the screen the time. It has two time setting options. The -e option sets the elapsed time and the -t option sets the alarm time. So lets say we are going to set it to alarm in 1 hour and 15 minutes. We will also set the foreground to red and the background blue. The line would look like:

alarm -f=1 -b=4 -e=1:15 &

The & is added to allow this program to run as a background task. You could also set the actual alarm time. Use the -t option.

There are a few caveats to be noted. First, as a background task, any screen output will be queued after a foreground task. So if the foreground task is idle, alarm's output will not appear. You will only see an output after some other screen response. If you want a pure alarm only, do not use the & option.

Second, I found that if something is going to the screen when alarm pops in, the current I/O may mix into the window. Depending on what you're running this could be a problem.

Finally, the alarm only checks the time on a minute by minute basis. Should something suspend the procedure for longer than a minute, the alarm time may be missed.

Next time I will show a different version using the system call ALARM. Also, I will try to include some program for experimenting with the color palette. Until then have fun!

```
LISTING 1
0001 /* ********************
0002
0003    File: alarm.c
0004    Date: 25 APR 89
0005    Author: Ron Voigts
0006    Compile: OS9: make
0007
0008    **********************
0009
0010    Function:
0011    1. alarm : signals user when specified
0012    time has elapsed.
0013    2. help() : provides help for pause().
0014    3. space( n ): prints n spaces.
0015    4. time_h(): extract hour from option argument.
0016    5. time_m(): extract minute from option argument.
0017    6. tcheck(): checks time.
0018    7. beep( n ): beeps terminal n times.
0019
0020    **********************
0021
0022    Verison 1.0    RDV
0023    25 APR 89
0024    Original
0025
0026    ********************** */
0027
0028 #include "getopt.h"
0029 #define FALSE 0
0030 #define TRUE 1
0031 #define STDOUT 1
0032 #define tps 100 /* For level 2 */
```

```
0033
0034 int win_flag;
0035 int f_ground;
0036 int b_ground;
0037 char c[1];
0038 char *opt;
0039 char *list="?wt=e=";
0040 int h, m;
0041 struct sgtbuf {
0042      char year,month,day,hour,minute,second;
0043      } t ;
0044 char answer[32];
0045
0046
0047 main( argc, argv )
0048 int argc;
0049 char **argv;
0050 {
0051
0052 /* Set up default values */
0053      win_flag=TRUE;
0054      f_ground=0;
0055      b_ground=1;
0056
0057 /* Get options */
0058      optn=1;
0059
0060      while( (opt=getopt( argc, argv, list )) != 0) {
0061           if ( opterr == -1 )
0062                printf("Illegal option %c\n", *opt );
0063           else {
0064                if (*opt=='?')
0065                     help();
0066                if ( toupper( *opt ) == 'F' )
0067                     f_ground=atoi( optarg );
0068                if ( toupper( *opt ) == 'B' )
0069                     b_ground=atoi( optarg );
0070                if ( toupper( *opt ) == 'W' )
0071                     win_flag=FALSE;
0072                if ( toupper( *opt ) == 'T' ) {
0073                     m=time_m( optarg );
0074                     h=time_h( optarg );
0075                }
0076                if ( toupper( *opt ) == 'E' ) {
0077                     m=time_m( optarg );
0078                     h=time_h( optarg );
0079                     getime( &t );
0080                     m+=t.minute;
0081                     if ( m>59 ) {
0082                          m-=60;
0083                          h+=1;
0084                     }
0085                     h+=t.hour;
0086                     if ( h>23 )
0087                          h-=24;
0088                     }
0089           }
0090      }
0091
0092 /* Returns when hour and minute have been reached */
0093      tcheck();
0094
0095 /* Send warning beeps */
0096      beep( 5 );
0097
0098      if (win_flag) {
0099
0100 /* Create an overlay window */
0101       owset(STDOUT, 1, 5, 10, 70, 4, f_ground, b_ground
);
0102
0103 /* Turn cursor off */
0104      curoff( STDOUT );
0105
0106 /* Print the option */
0107      printf("\n");
0108      space( 27 );
0109    printf("The Time Is %02d:%02d\n",  t.hour, t.minut
);
0110      space( 25 );
0111      printf(" Type <Enter> To Continue!\n");
0112      gets( answer );
0113
0114 /* Turn cursor on */
0115      curon( STDOUT );
0116
0117 /* End the overlay window */
0118      owend( STDOUT );
0119
0120      } else {
0121
0122     printf("The Time Is %02d:%02d\n", t.hour, t.minut
);
0123      printf("Type <Return> To Continue!\n");
0124      gets( answer );
0125
0126      }
0127
0128 }
0129
0130
0131 /* Print help information */
0132 help()
0133 {
0134       printf("Alarm [-f=n] [-b=m] [-w] [-e=hh:mm] [-
t=hh:mm]\n");
0135      printf("      e, elapsed time to alarm\n");
0136      printf("      t, time of alarm\n");
0137    printf("    f, changes foreground color to n (defaul
0)\n");
0138    printf("    b, changes background color to m (defaul
1)\n");
0139      printf("      w, turns off window\n");
0140      exit(0);
0141 }
0142
0143 /* Return the hour from option argument */
0144 int time_h( s )
0145 char *s;
0146 {
0147      return( atoi( s ) );
0148 }
0149
0150 /* Prints n spaces */
0151 space( n )
0152 int n;
0153 {
0154      while ( n-->0 )
0155           printf(" ");
0156 }
0157
0158
0159
0160 /* Return the minute from option argument */
0161 int time_m( s )
0162 char *s;
0163 {
0164      while ( *s++!=':' )
0165           if ( *s=='\0' )
0166                return( -1 );
0167      return( atoi( s ) );
0168
0169  }
0170
0171 /* Checks the time.  Returns on h and m match */
0172 tcheck()
```

```
0173 {
0174     do {
0175          getime( &t );
0176     } while ( t.hour!=h || t.minute!=m );
0177 }
0178
0179 /* Beeps terminal n times */
0180 beep( n )
0181 int n;
0182 {
0183      while ( n- )
0184           printf("%c", '\007');
0185 }
0186
```

LISTING 2

```
0000 /* *********************
0001
0002    File: cgfx.c
0003    By: Ron Voigts
0004    Date: 29 FEB 89
0005
0006    **********************
0007
0008    Fucntion:
0009    1. owset() : open an overlay window
0010    2. owend() : close window
0011    3. curon() : cursor on
0012    4. coroff(): cursor off
0013
0014    ***********************
0015
0016    Version 1.0       RDV
0017    19 FEB 89
0018    Original
0019
0020    ********************* */
0021
0022 /* Creates an overlay window of
0023    size szx x szy starting at position
0024    cpx, cpy.  Foreground and background
0025    colors are fprn and bprn.  If svs is 0,
0026    area under window is not save, if it is
0027    1 it is saved.                          */
0028
0029 owset(path,svs,cpx,cpy,szx,szy,fprn,bprn)
0030 int path,svs,cpx,cpy,szx,szy,fprn,bprn;
0031 {
0032
0033 /* Create buffer for 'write' string */
0034      char b[9];
0035
0036 /* Initialize the buffer */
0037      b[0]=0X1B;
0038      b[1]=0X22;
0039      b[2]=svs;
0040      b[3]=cpx;
0041      b[4]=cpy;
0042      b[5]=szx;
0043      b[6]=szy;
0044      b[7]=bprn;
0045      b[8]=fprn;
0046
0047 /* Write the buffer */
0048      if ( write( path, b , 9 ) == -1 )
0049           return( -1 );
0050
0051      return( 0 );
0052
0053 }
0054
0055
0056 /* Closes an existing window */
0057 owend( path )
0058 int path;
0059 {
0060
0061 /* Create buffer for 'write' string */
0062      char b[2];
0063
0064 /* Initialize buffer */
0065      b[0]=0X1b;
0066      b[1]=0X23;
0067
0068 /* Write the buffer */
0069      if ( write( path, b, 2 ) == -1 )
0070           return( -1 );
0071
0072      return( 0 );
0073 }
0074
0075
0076 /* Turns cursor on */
0077 curon( path )
0078 int path;
0079 {
0080
0081 /*  Create buffer for 'write' string */
0082      char b[2];
0083
0084 /* Initialize buffer */
0085      b[0]=0X05;
0086      b[1]=0X21;
0087
0088 /* Write the buffer */
0089      if ( write( path, b, 2 ) == -1 )
0090           return( -1 );
0091
0092      return( 0 );
0093 }
0094
0095
0096 /* Turns cursor off */
0097 curoff( path )
0098 int path;
0099 {
0100
0101 /* Create 'write' buffer */
0102      char b[2];
0103
0104 /* Initialize buffer */
0105      b[0]=0X05;
0106      b[1]=0X20;
0107
0108 /* Write the buffer */
0109      if ( write( path, b, 2 ) == -1 )
0110           return( -1 );
0111
0112      return( 0 );
0113 }
```

LISTING 3

```
0000 /* *************************
0001
0002    Name: GETOPT
0003    By: Ron Voigts
0004    Date: 25-MAY-87
0005
0006    ***************************
0007
0008    Function:
0009    This function examines the argument list
0010    returning a pointer to the option and
0011    its argument.  A null string is pointed
0012    to if the option has now argument.
0013
```

```
0014    ****************************
0015
0016    Version 1.00
0017    Original.
0018
0019    ************************** */
0020
0021 #define TRUE 1
0022 #define FALSE 0
0023
0024 char *optarg; /* Option argument */
0025 int optn;     /* Next option     */
0026 int opterr;   /* Error status    */
0027
0028 char *getopt( c, v, optlist )
0029 int c;          /* argument count  */
0030 char **v;       /* argument vector */
0031 char *optlist; /* option list     */
0032
0033 {
0034    int isoption;     /* option flag  */
0035    int hasarg;       /* option argument flag */
0036    register int i;   /* useful index */
0037    char *opt;        /* option pointer */
0038    char *t;          /* argument pointer */
0039    static char *null='\0'; /* null string */
0040
0041 /* Set up the null string for 'optarg' */
0042    optarg = null;
0043
0044 /* Set up the error return status */
0045    opterr = 0; /* No errors */
0046
0047 /* Set up the argument */
0048    t=v[optn];
0049
0050 /* We are at the end of the argument list */
0051       if ( (optn==c) || (*t!='-') || ( *t=='-' &&
*(t+1)=='\0' ) )
0052          return( 0 );
0053
0054 /* We can set the option */
0055    opt = t+1;
0056
0057 /* Check if we have an option with an argument */
0058    isoption = FALSE;
0059    hasarg = FALSE;
0060    for ( i=0; i<strlen(optlist); i++ )
0061       if ( toupper(*(t+1))==toupper(optlist[i]) ) {
0062          isoption = TRUE;
0063          if ( optlist[i+1]==':')
0064             hasarg = TRUE;
0065       }
0066
0067 /* If this is not an option then return with error *
0068    if ( !isoption )
0069       opterr=-1; /* illegal option */
0070
0071 /* Now we check and set up the argument */
0072    if ( hasarg ) {
0073       if (*(t+2) == '\0')
0074          if ( optn < c-1 )
0075             optarg = v[++optn];
0076          else
0077             opterr=-2; /* Missing option argument *
0078       else
0079          optarg = t+2;
0080       if ( *optarg==':' )
0081          optarg++;
0082    } else
0083       if ( *(t+2) != '\0' )
0084          opterr=-3; /* Argument not expected */
0085
0086 /* Now we have an argument and option */
0087    optn++; /* Adjust the next pointer */
0088    return( opt ); /* Return the option pointer */
0089
0090 }
0091
```

LISTING 4

```
0000 extern char *optarg; /* Option argument */
0001 extern int optn; /* Next option */
0002 extern opterr; /* Error Status */
0003
```

+++

FOR THOSE WHO NEED TO KNOW
68 MICRO JOURNAL™

C User Notes

The C Programmers Reference Source. Always Right On Target!

A Tutorial Series

By: Dr. E. M. 'Bud' Pass
1454 Latta Lane N.W.
Conyers, GA 30207
404 483-1717/4570
Computer Systems Consultants

INTRODUCTION

On a multi-user system, especially one which allows users to select their own passwords, the use of passwords which are in dictionaries or are otherwise obvious can seriously compromise system security. This chapter provides a function, provided by John Nagle, which tests prospective passwords to ensure that they are not potentially common English words.

PASSWORD TESTING

Given a free choice, altogether too many users choose passwords which may be easily guessed. Dennis Richie, in his "Notes on the Security of UNIX", comments that with passwords the only line of defense in many systems, it would be desirable to prevent users from choosing ones that leave the system wide open to any cracker.

This small function uses a little-known property of the English language to detect candidate passwords that might be easy to guess. The function might be built into the password-changing utility function of a system, so that all new passwords must be considered non-obvious to be accepted.

The algorithm depends upon a subtle property of English. Less than one-third of the possible "triples", sequences of three letters, are used in English words. This property makes it possible to distinguish random letter strings from strings that look like English words. The word "password", for example, contains the five triples "pas", "ass", "ssw", "wor", "ord".

All five of these triples, therefore, are used in English. The triple "xqy", on the other hand, appears in no common English word. In general, a triple chosen at random has only one chance in three of appearing in any English word. Starting with a suitable large list of words, such as a dictionary, we can make a table of all the triples that appear in the list of words. We can then test words against the table by extracting all triples from the word and looking up the triples in the table. If the word contains several triples that are not in the table, it is almost certainly not an English word, and definitely non-obvious.

The table of triples seems at first unwieldy, but a compact representation is possible. The table in the subroutine is essentially a 3-dimensional Boolean array, 27 x 27 x 27. There are thus 19,683 slots in the table, each containing one bit. C does not provide a built-in representation for packed Boolean arrays, so the third dimension of the array is handled by using a "long" value for each group of 27 bits. Letters are mapped to the range 1..27, so that "a" or "A" is represented by 1, "b" or "B" by 2, and so forth. Non-letters are mapped to zero. For every possible sequence of three letters, then, there is a unique bit in the table. That bit is a 1 if the three letter sequence is used in English. So we can take any sequence of three letters, look it up in the table, and find out if it is a triple known to be used in the English language.

The triple "pas", for example, maps to triple number (16,1,19). Array element [16,1] in the table is hex 07fffabc, and bit 19 of that value is a 1. So, the triple "pas" is known to be used in some existing word, and the odds are that the word from which the triple was drawn is an English word or looks like one.

The table was built with a program that extracted all the triples in the UNIX spelling dictionary and set the appropriate bit for each triple. Along with the UNIX list of words, a few other obvious patterns were thrown in; the sequences "aaa", "bbb", and so forth, the alphabet, and the rows of the "qwerty" typewriter keyboard. Building the table is a straightforward process, and, with a machine-readable dictionary or just a large body of text to use as raw data, you can write your own table-builder and build a table of your own. Any table, though, based upon a list of English words, will be very similar to the one given here. The triple statistics are a real property of English, not an artifact of the word list used.

This is definitely a detector for obvious English words. Words in other languages, particularly ones distant from English, often pass. "Bejing" and "Timbuktu" are considered non-obvious.

The test considers any word with at least two triples not found in the table to be non-obvious. This makes the odds quite good that a randomly chosen string of letters will pass and be considered non-obvious, and thus a suitable password. More than 95% of all eight-letter sequences chosen at random will pass. Even for a five-letter sequence, the minimum considered a good defense against trying all possibilities, most randomly chosen sequences will pass. But every word in the UNIX dictionary, and almost all English words generally, will be rejected as obvious.

This technique isn't idiot-proof. It is possible, with effort, to come up with an easily-guessed password that will pass the test. But it's more work than coming up with a good password.

OBVIOUS C FUNCTION

Following is a C function which implements Nagle's ideas just described.

```
/*

Obvious password detection subroutine.

Function:

char *obvious(word)
char word[];


Returns "ok" if password is acceptable.
Returns a pointer to a message if the
password is unacceptable.

The algorithm used requires that the
length of the password be within config-
urable length limits, and that the pass-
word not have triplet statistics similar to
those associated with words in the
English language.  This is an inversion of
a technique used to find spelling errors
without a full dictionary.  No word in the
UNIX spelling dictionary will pass this
algorithm.

Users should be advised to pick a
password composed of random letters
and numbers.  Eight randomly chosen
letters will pass the algorithm over 95%
of the time.  A word prefaced by a digit
will not pass the algorithm, although a
word with a digit in the middle usually
will.  Two words run together will often
pass.

John Nagle
Ford Aerospace and Communications
Corp.
Western Development Laboratories
3939 Fabian Way
Palo Alto, CA  94303


*/

/*

Table of triple usage in text
24511 words were used to make this table.
The words came from the files:

/usr/dict/words
obvpats.lp

The table is 30 percent populated.

*/

#include <stdio.h>

long obvtab[27][27] =
{

{

0x00100001, 0x00040000, 0x00040000,
0x00009000, 0x00808020, 0x00140000,
0x00000000, 0x00080000, 0x00000002,
0x00000000, 0x00000000, 0x00000000,
0x00001020, 0x00000000, 0x00000010,
0x00000020, 0x00000000, 0x00000000,
0x00000030, 0x00300100, 0x00000100,
0x00000000, 0x00000020, 0x00000000,
0x00000000, 0x00000000, 0x00000000

},
{

0x00090000, 0x000c708a, 0x022cd73e,
0x023ffbbe, 0x02fffffe, 0x002cd0da,
0x023cdae2, 0x02adf3b6, 0x0024f222,
0x00dd7efa, 0x00008022, 0x02b5937a,
0x06fdfbfe, 0x02bdf37e, 0x07fffdfff,
0x003c3248, 0x023dfb76, 0x00200000,
0x07fffffe, 0x02bbfbbe, 0x06bcfb6f,
0x055f7dfc, 0x02609232, 0x021cfbf6,
0x02999222, 0x00bdf3ff, 0x06308232

},
{

0x00080000, 0x07fdfbfc, 0x022c9226,
0x00008012, 0x00248222, 0x07dffbfe,
0x00200000, 0x00000000, 0x00208020,
0x047ffdfe, 0x00000020, 0x00000000,
0x02208222, 0x00800220, 0x0000a022,
0x03fdfffe, 0x00048000, 0x00000000,
0x02208222, 0x00308228, 0x00049062,
0x071efefc, 0x00000220, 0x00000020,
0x00000000, 0x049d11a2, 0x00000000

},
{

0x00080080, 0x02fffffe, 0x000c0020,
0x0024932a, 0x0000802a, 0x06bdf3fe,
0x00000002, 0x00248220, 0x02bdf3fe,
0x007df0fe, 0x00000000, 0x02bdfffe,
0x02208222, 0x00200222, 0x00008222,
0x07fffffe, 0x00000102, 0x00200000,
0x02308222, 0x00188200, 0x02aca262,
0x003df27e, 0x00000004, 0x00000000,
0x00000000, 0x001dd08e, 0x00000022

},
{

0x00088022, 0x06fd7bfc, 0x02249222,
0x00209107, 0x02249332, 0x07fff7fe,
0x00249282, 0x02a43a22, 0x00008222,
0x05fdf3fe, 0x00208022, 0x00000220,
0x02208222, 0x00208222, 0x00008223,
0x07fdf3fe, 0x00049322, 0x00200000,
0x02208222, 0x0091bb2a, 0x00008302,
0x043dfafe, 0x00008222, 0x02048222,
0x00000000, 0x00987aef, 0x00200000

},
{

0x004c1030, 0x04ff79dc, 0x023c9226,
0x023c9b3a, 0x02bdf3fe, 0x04ddfbfe,
0x023c92fa, 0x0224f3e6, 0x0224a222,
0x045d7afa, 0x00208022, 0x0218ca36,
0x02fdfbfe, 0x02a9f22e, 0x06ffffff,
0x05fd73d8, 0x02bdb32e, 0x00200000,
0x07ffffe, 0x02fbfbbe, 0x06bdfb6e,
0x005d78bc, 0x066c8222, 0x029df37e,
0x0333832a, 0x008df97f, 0x06648222

},
{

0x00080000, 0x07bc7bfe, 0x00000222,
0x0000000a, 0x00008002, 0x02fc7abe,
0x022cb366, 0x00000102, 0x00000022,
0x055c70fe, 0x00008000, 0x00000002,
0x02208222, 0x0000000a, 0x00000200,
0x03edf2be, 0x00000008, 0x00000000,
0x02208222, 0x00118922, 0x028ca32e,
0x041c74b8, 0x00000000, 0x00000002,
0x00000000, 0x00000000, 0x00000000

},
{

0x00080000, 0x06fdf3fc, 0x00008222,
0x00000000, 0x00008022, 0x02fdf3fe,
0x00240220, 0x020d93a2, 0x02bddffe,
0x047df0fe, 0x00000002, 0x00008200,
0x02208222, 0x02300222, 0x00318226,
0x02fdf3fe, 0x00009220, 0x00000000,
0x02208222, 0x00919b06, 0x00048302,
0x061cf2ee, 0x00000000, 0x02048022,
0x00000000, 0x00056002, 0x00000002

},
{

0x00080000, 0x06fdfbfe, 0x02249222,
0x00048102, 0x00048202, 0x07fff2ff,
0x00208202, 0x00008022, 0x00008122,
0x047dfcfe, 0x00000800, 0x00008220,
0x02248222, 0x0008a222, 0x0028c222,
0x06fdfeff, 0x00209202, 0x00200000,
0x02249232, 0x00108302, 0x02bdf3fa,
0x071d71fe, 0x00000200, 0x00008322,
0x00000000, 0x011d32be, 0x00000000

},
{

0x00483010, 0x05dd7bbc, 0x022cf226,
0x023cdb2a, 0x02bdf6b6, 0x04fd79fc,
0x0234d262, 0x063df3a2, 0x00000222,
0x00000202, 0x00208002, 0x02208a22,
0x02fdbbfe, 0x022df36e, 0x07ffffff,
0x05fd70be, 0x02bdb766, 0x00200000,
0x02fdfafe, 0x02ffffff, 0x06bdfb6f,
0x000c7008, 0x02248222, 0x00008202,
0x023082e2, 0x00000002, 0x04608022

},
{

0x00080000, 0x07ed7bbc, 0x00000000,
0x00000000, 0x00000000, 0x009cd57a,
0x00000000, 0x00000000, 0x00000000,
0x005070c4, 0x00000400, 0x00001000,
0x00000000, 0x00000000, 0x00000000,
0x02fc5bae, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000,
0x015d7e96, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000

},
{

0x00080000, 0x07fdfff4, 0x00248222,
0x00040302, 0x00048022, 0x02fdf3f6,
0x00208222, 0x00048000, 0x0014a022,
0x00ddf8fe, 0x00000002, 0x0000ca22,
```

```
0x0220a222, 0x00000022, 0x00208222,
0x01fdfb7a, 0x00009202, 0x00000000,
0x02208222, 0x02b1abae, 0x00048302,
0x029d7110, 0x00000000, 0x00008322,
0x00000000, 0x00ac9522, 0x00000000

},
{

0x00088000, 0x07fffbfe, 0x02249222,
0x0234b322, 0x02bdf36e, 0x07fdfffe,
0x00bc93e6, 0x00048222, 0x00008222,
0x05fffdfe, 0x00008000, 0x0288d322,
0x02bdb7fe, 0x0228e332, 0x00208020,
0x07fffffe, 0x021ca362, 0x00000000,
0x02208022, 0x0299ab26, 0x06eca326,
0x055dfefe, 0x00008222, 0x02008322,
0x00000000, 0x00bdfbfe, 0x00000022

},
{

0x00080000, 0x07fdffbc, 0x002dd236,
0x0021fbfe, 0x00008020, 0x07ddfbfe,
0x00209220, 0x00000000, 0x00208020,
0x053efcfa, 0x00000000, 0x00000000,
0x00008222, 0x0220a222, 0x02208222,
0x07fdfbfe, 0x023c9be6, 0x00200000,
0x00088002, 0x0239a37c, 0x00048120,
0x041dfaf8, 0x00000020, 0x00008222,
0x00000000, 0x001cd0a8, 0x00000000

},
{

0x00180000, 0x06fdfbfe, 0x002c922a,
0x02349b22, 0x02bffffe, 0x07fff3ff,
0x00249222, 0x02bdbb76, 0x02208222,
0x057ffbfe, 0x00208022, 0x021d33ea,
0x02008222, 0x00008222, 0x0228c222,
0x07fdf2fe, 0x00209220, 0x00200000,
0x0220822a, 0x02fdfbfe, 0x06adf3ee,
0x043dfafe, 0x02208222, 0x00048322,
0x00100200, 0x01ffb11e, 0x02808222

},
{

0x0008113c, 0x017f79da, 0x02fcd77e,
0x02349b2a, 0x02adfbfe, 0x037ef35c,
0x0234b262, 0x02bcf7f6, 0x0204f222,
0x011c58fe, 0x00008222, 0x02a99b2e,
0x06fdfbfe, 0x02aff36e, 0x07ffffff,
0x045df9dc, 0x02bffbae, 0x00200000,
0x06fffbfe, 0x02fbbbae, 0x02bdf3ee,
0x055f73ff, 0x02008a22, 0x06bdfb7e,
0x02d043ea, 0x001dd37e, 0x06008222

},
{

0x00080000, 0x07fffabc, 0x002c9022,
0x00008100, 0x00040002, 0x07fdfefe,
0x00241220, 0x00040002, 0x023cd232,
0x057ff8fa, 0x00000002, 0x00000220,
0x02208222, 0x00208022, 0x00008020,
0x03fdfafa, 0x022db322, 0x00040000,
0x02208222, 0x02b0bb2a, 0x0224832a,
0x041d7afe, 0x00000008, 0x00048202,
0x00000000, 0x00949b86, 0x00000000

},
{

0x00080000, 0x00100000, 0x00000000,
0x00000000, 0x00000000, 0x00000010,
0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00020000,
0x00080000, 0x00000000, 0x00000000,
0x02008222, 0x00000000, 0x00000020,
0x00000000, 0x00000000, 0x00000000

},
{

0x00080010, 0x07ffffffc, 0x022c9222,
0x023c9b2a, 0x02bdd366, 0x07ffffff,
0x00249222, 0x022cb322, 0x02208222,
0x07fbfdfe, 0x00208020, 0x029df336,
0x02b98236, 0x0239936e, 0x02b9ab66,
0x07ffffbe, 0x023cb322, 0x00200000,
0x02248322, 0x02f3fb7e, 0x06bcf3ee,
0x01ddf2fe, 0x02208222, 0x00008222,
0x00000020, 0x0099f0be, 0x00008222

},
{

0x00081000, 0x03fdfffe, 0x02248222,
0x0224b323, 0x00248242, 0x07fffbfe,
0x02208222, 0x002c8220, 0x02f5fa6e,
0x057ff8fe, 0x00200000, 0x0224a222,
0x02208222, 0x02208222, 0x02208223,
0x06f5f7fe, 0x022c9326, 0x00200000,
0x00208222, 0x02bdf36e, 0x02edf7ff,
0x041df2fe, 0x00010220, 0x00248222,
0x00000000, 0x001d78bc, 0x00000000

},
{

0x001c1010, 0x03fdfbfc, 0x00248222,
0x02249102, 0x00048000, 0x03fdfbfe,
0x00249222, 0x00248022, 0x02bffbfe,
0x047ff8fe, 0x00000000, 0x02000222,
0x02208222, 0x00208222, 0x00308223,
0x07fdfafe, 0x00049222, 0x00000000,
0x02a08222, 0x02f1ab7e, 0x027c9be2,
0x031df2fe, 0x00000202, 0x01208322,
0x00000000, 0x01ad328e, 0x000db2a2

},
{

0x004c1010, 0x02dd7bdc, 0x027db736,
0x023c9b2a, 0x06ac92b6, 0x063df5f6,
0x0030b042, 0x002cf3a2, 0x00004002,
0x057dd0be, 0x00208202, 0x00048a30,
0x06fdfafe, 0x007bf36e, 0x02bdfbfe,
0x023d5210, 0x02bdbbfe, 0x00200000,
0x06fffbfe, 0x023bdbbe, 0x06bcf3be,
0x00202000, 0x00840220, 0x00000002,
0x00309220, 0x00601022, 0x04208022

},
{

0x00080000, 0x003d7fbc, 0x00004000,
0x00000000, 0x00000002, 0x039dfbfe,
0x00000000, 0x00000000, 0x00000000,
0x057cdafe, 0x00000000, 0x00000100,
0x00008002, 0x00000000, 0x00000000,
0x02bcfa88, 0x00000000, 0x00000000,
0x00008020, 0x00000800, 0x00000000,
0x000c1000, 0x02400220, 0x01000000,
0x00000000, 0x00800200, 0x00008000

},
{

0x00080000, 0x077d7bde, 0x00248222,
0x00208102, 0x02248220, 0x025d53bf,
0x00209202, 0x00000200, 0x02208222,
0x055d70f8, 0x00000000, 0x02804200,
0x02180222, 0x00000222, 0x02bd03fc,
0x00fdf864, 0x00209222, 0x00000000,
0x02008222, 0x02bdbb2c, 0x00208322,
0x00057100, 0x00000000, 0x00808002,
0x02000000, 0x0000f022, 0x02000000

},
{

0x00080000, 0x005870d8, 0x00000000,
0x00649322, 0x00000000, 0x021c7298,
0x00008000, 0x00201000, 0x00208202,
0x0058f0fe, 0x00000000, 0x00000000,
0x00000020, 0x00000000, 0x00000002,
0x001d6090, 0x00249222, 0x00200000,
0x00000000, 0x00000020, 0x02248326,
0x00051016, 0x00000200, 0x00008020,
0x01008000, 0x04041080, 0x00000000

},
{

0x004c1010, 0x0097f998, 0x00248232,
0x00009322, 0x0004c226, 0x00ddf0f6,
0x00201222, 0x0224f230, 0x02048020,
0x00094030, 0x00000002, 0x00004420,
0x00699a22, 0x0221e22e, 0x0310c3ba,
0x04fd78d8, 0x023dd322, 0x00200000,
0x0234c272, 0x0039b32e, 0x02108322,
0x00095b88, 0x00000022, 0x0004832a,
0x00000200, 0x02000000, 0x00000002

},
{

0x00080000, 0x00557b8e, 0x00000000,
0x00000000, 0x00000002, 0x00bc5a3e,
0x00000000, 0x00000020, 0x00000000,
0x003df0ae, 0x00000000, 0x00000000,
0x02008020, 0x00000002, 0x00000000,
0x0065e2b2, 0x00000002, 0x00000000,
0x00008002, 0x00000008, 0x00000020,
0x00040820, 0x00008020, 0x00008020,
0x00000008, 0x000002084, 0x06809222

}

};

/*

Configuration parameters

*/
#define MINLENGTH 5 /* minimum
password length */
#define MAXLENGTH 8 /* maximum
password length */
#define MINNOFIND 2 /* minimum
unusual triples */

int usual;          /* count of usual
triples */

/*

dotriple  -  called by obvword

If this triple is not used by any
word used
to build the table, we tally that
fact.

*/
dotriple(m1, m2, m3)
short m1;           /* all in 0..26
*/
short m2;
short m3;
{

usual += (!(obvtab[m1][m2] & (1L <<
m3)));

}
```

```
/*

obvword  -  do one word

dotriple is called on each 3-character triple in the
word,
using a mapped value of the character into the range
0..26,
where letters map into 1..26 regardless of case and
everything
else maps to zero.


*/

obvword(word)
char word[];
{

int i;          /* for loops */
int patcnt = 0; /* count in word */
char ch;        /* working char */
short pat[32];  /* pattern of mapped values */

for (usual = i = 0; (word[i] && (i < sizeof(pat))); ++i)
{

patcnt = i;     /* max value */
ch = word[i];   /* get character */
if ((ch >= 'a') && (ch <= 'z'))

pat[i] = ch + 1 - 'a';

else
if ((ch >= 'A') && (ch <= 'Z'))

pat[i] = ch + 1 - 'A';

else

pat[i] = 0;     /* map into 0..26 */

}
/* for all triples */
for (i = 0; i < (patcnt - 1); ++i)

dotriple(pat[i], pat[i + 1], pat[i + 2]);

return (usual);

}

/*

obvious  -  check word for obviousness

Words are rejected for being too short or
too long, or looking like English words.

*/
char *obvious(word)
char word[];
{

int i;

if ((i = strlen(word)) < MINLENGTH)

return("too short");

if (i > MAXLENGTH)

return("too long");

if (obvword(word) < MINNOFIND)

return("too obvious");

return("ok");

}
```

EXAMPLE C PROGRAM

Following is this month's example C program; it provides a program to enable the testing of prospective passwords from a terminal, using the obvious function described above.

```
main()
{

char pword[32];

printf("Password obviousness tester\n");
for ( ; ; )
{

printf("Enter password: ");
if ((!fgets(pword, 32, stdin)) || (!*pword))

break;

*(pword + strlen(pword) - 1) = 0;
printf("- %s\n", obvious(pword));

}
exit(0);

}

+++
```

FOR THOSE WHO NEED TO KNOW 68 MICRO JOURNAL™

FORTH

A Tutorial Series

By: R. D. Lurie
9 Linda Street
Leominister, MA 01453

FORTHBUILDER

Wilson Federici has come up with another software gem! I have had several months to experiment with FORTHBUILDER, and, on a scale of 1-10, I would rate it at least a 12!
See page 31 this issue South East Media Catalog!!!

FORTHBUILDER is a cross-compiler for FORTH. This means that FORTHBUILDER is not just another run-of-the-mill FORTH, but is a compiler which is used to generate FORTH compilers to run on target machines. In other words, you use FORTHBUILDER to generate a FORTH system to run on some other machine, the CoCo, for example.

FORTHBUILDER will run under either of two host systems, FLEX or MS-DOS; you must specify which one you want when you order. FORTHBUILDER generates code for either of two target cpu's, the 6809 or the 6502; again, you must specify which one you want.

No matter what you order, you get a disk with the FORTHBUILDER program and a number of files which can be used as is, or as models, to generate a full 1983 standard FORTH. The files can be used to create a FORTH system to run completely in RAM or to run completely in ROM (except for a minimal RAM requirement).

With a little imagination, you can edit these files to produce any system you want. You can even clip out all of the unnecessary words for a particular application, so that you can cut the memory requirement to a minimum. This way, you might well generate a program which would be shorter than you would ordinarily write for the same job in Assembly language (yes, this is possible, and one of the distinct virtues of FORTH).

You can add words to the files used to generate the resulting FORTH, so that it would not be necessary to compile additional words to add to the FORTH generated by FORTHBUILDER. This is a little hard to state clearly, so maybe I had better explain a little more. If you simply compile an 83-FORTH from the model provided, you would still have to compile in additional words to accomplish a particular job. However, if you edit the model to add these words to the original source files, you would not then have to compile any more words in order to have a functioning application for a particular job.

Incidentally, if you are interested in maintaining the secrecy of your source code, just set the FORTHBUILDER option WIDTH to 0. This will cause the generated code to have no headers; thus it would be very uneconomical at best and, more likely, virtually impossible to reverse engineer! This would not stop piracy, but it sure ought to slow it down!

Just because I used the version of FORTHBUILDER supplied to run on FLEX, I am not limited to writing FORTHs for FLEX systems. Mainly, I have concentrated on using FORTHBUILDER to create a functional duplicate of Federici's FF9 to run directly on the CoCo3. I simply edited the model code for KEY , ?KEY , and EMIT to use the CoCo3 ROM. I also adjusted the disk parameters to fit a disk formatted by that ROM and hook onto the ROM disk calls so that I could completely eliminate the need for FLEX. I hope to put this FORTH, which I call C3FORTH, out for beta-testing very soon. If all goes well, I plan to release it to the public domain via CompuServe, DELPHI, and Genie. (At this stage, I don't plan to distribute it on disk; that is too much hassle.)

The listing in Figure 1 shows how easy it is to use FORTHBUILDER to do this. Notice that this is a FLEX text file named "COCO.TXT" which is called by the FLEX command:

FB6809 COCO.TXT

Everything else is pretty much automatic from here. Each of the seven source files are called in order and compiled, a transfer address is assigned, the binary object file named "CF.BIN" is saved to disk, and execution is terminated.

```
CR .( F9_0.TXT)
SOURCE.FILE F9_0.TXT
CR .( F9_1.TXT)
SOURCE.FILE F9_1.TXT
\ LIST.UNDEFINED
CR .( F9_2.TXT)
SOURCE.FILE F9_2.TXT
\ LIST.UNDEFINED
CR .( F9_3.TXT)
SOURCE.FILE F9_3.TXT
\ LIST.UNDEFINED
CR .( F9_4.TXT)
SOURCE.FILE F9_4.TXT
\ LIST.UNDEFINED
CR .( F9_RDL.TXT)
SOURCE.FILE F9_RDL.TXT
\ LIST.UNDEFINED
CR .( F9_IO.TXT)
SOURCE.FILE F9_IO.TXT
\ LIST.UNDEFINED
CR .( F9_5.TXT)
SOURCE.FILE F9_5.TXT
LIST.UNDEFINED
4096 TRANSFER.ADDRESS
OBJECT.FILE CF.BIN
EXIT.COMPILER
```

Figure 1. A command file used with FORTHBUILDER.

The files named "F9_0" through "F9_5" are those supplied as models on the original FORTHBUILDER distribution disk. I

made a few minor changes in a couple of these files in order to change the transfer address and the I/O calls. "F9_RDL" has the disk changes and "F9_IO" has the changes to KEY, ?KEY, and EMIT. These two files could have been included within one of the supplied files, but I made them separate for more convenient editing; they went through several versions before I was happy with them.

Since FORTHBUILDER treats this file as a series of FORTH commands, the \ can conveniently be used to edit this file by changing commands into comments. The LIST.UNDEFINED command is a debugging aid which helps you to find where an undefined word is first called. At first, I used this command to follow the action of the compiler, but later got tired of all of the extra display, so I just commented it out everywhere except at the end, where it should always be present.

FORTHBUILDER allows forward references so that you could leave out a definition after a lengthy series of editing sessions. If this happens, you would get an error signal, and LIST.UNDEFINED would help you pinpoint the problem.

The lines containing the .(just show on the screen the currently compiling file. This is just to keep me from getting antsy while the program is running, but I don't know what it is doing. You could leave out these lines without doing any harm, but I like to follow the action of a long program; it keeps me from getting bored!

There is no trouble with error signals, but FORTHBUILDER is so easy to use that you may never see one! I could not manufacture a situation in which I did not get the appropriate error trapping and compiler action, so I am confident that the user is quite well protected. On the other hand, I am also sure that you could screw up if you really tried to trick the compiler, so don't get cute!

The instruction manual which comes with FORTHBUILDER is entirely adequate, provided you understand how to use the host DOS. You must also have a good working knowledge of FORTH in order to go beyond the supplied model. I have not counted the pages in the instruction manual (which seems to be the thing to do, why??), but I assure you that everything you need to know is there. On the other hand, you may have to read it a couple of times before you can appreciate some of the subtleties of FORTHBUILDER.

FORTHBUILDER is written in FORTH, so you can even extend the cross-compiler, though I certainly don't recommend that you try. For that matter, I can't imagine why you would want to. However the manual does tell you how.

All-in-all, if you have a need to generate your own version of FORTH, I don't know how you could do better than with FORTHBUILDER. At the price, you could probably even buy a CoCo3, FLEX, and FORTHBUILDER and still come out cheaper than buying some of the other cross-compilers for FORTH on the market.

FORTHBUILDER, $99.95
South East Media
5900 Cassandra Smith Rd.
Hixson, Tn. 37343
Tele. (615) 842-4600
FAX (615) 842-7990

TIPS FOR BEGINNERS

Selecting the Value Stored in BASE

At first, I was very confused about selecting a value for BASE, especially when I was editing a set of screens for later compiling. Of course, the obvious choice is DECIMAL, since that is the default. However, it was also obvious that I needed to change to HEX if I wanted to process numbers in base-16. But it is sometimes very convenient to switch back and forth among several bases during the course of executing a program; so how and when do I do it?

In all versions of FORTH that I know of, DECIMAL and HEX come predefined, but you should also define other bases you expect to work in, just to help keep track of program flow. After all, you can't print a number in decimal or hexadecimal if you are currently in base-2 or base-8, etc.

As an example, look at lines 4 and 5 of the listing for screen #1. Here I have defined BIN, which is the definition for a word to convert the base to binary (base-2). Any other base, such as octal (base-8) or septal (base-7) can be defined in exactly the same way. In fact, any base can be used, up through 72! Go ahead and see what happens when you try to print all 72 characters in base-72.

Now that BIN has been defined, let's see how to use it inside a definition. The word .BIN defined in lines 7-10 of the same screen show one way that it could be used. The purpose of .BIN is to print each bit of a 32-bit number as either a "0" or a "1", which is the proper way to print a binary number, since those are the only two digits available in binary. Ignore the rest of the definition for now; I'll talk more about it in another column.

The first thing that happens when .BIN is called is that the value in BASE is changed to 2, which switches the system to binary numbers. The rest of the definition is then executed in that base. Notice that I did not bother to change out of binary before exiting from .BIN, since I use the convention that it is the responsibility of any definition to set its own value into BASE, if that is important to the definition. This convention keeps me from having to worry about whether or not a definition has the proper base to work with; it just sets its own value.

The definition of DEMO1 shows when the confusion with changing bases can overwhelm you. First of all, I found it most convenient to write DEMO1 in HEX, because the number $7FFFFFF7 was contained within the definition. This number is certainly easier to type in this form than in its decimal form of 2147483639. This long decimal number is easy to mis-type; in fact, I just did so, and wasted a lot of time in verifying that I had finally got it right (I hope!). The rest of the definition is also in base-16, so the "10" in line 3 is equivalent to 16 in decimal and the "0C" in line 6 is equivalent to 12 in decimal.

****** RULE #1: All numbers within a compiling definition must be related to the value stored in BASE at the time of compilation. *****

Since I wanted the resulting number to be printed in decimal form, it was necessary to include the DECIMAL in line 6 to

insure this, because the .BIN in line 8 causes a shift to binary with each passage through the loop. Therefore, I must shift back and forth between base-10 and base-2 in order to get the printed output that I want. A small portion of the output form DEMO1 is shown in Figure 2.

I exited the compilation of DEMO1 still in base-16. Here I change conventions. Unless there is a compelling reason to do otherwise, I always change the base back to DECIMAL before I quit compiling a screen. That way, I always know that I am in the default condition and don't waste a lot of time trying to figure out why the numbers that I type in after compiling one or more screens don't always give me the same result. For example, simply commanding 16 LIST produces completely different and aggravating results, depending on whether I am in DECIMAL or HEX when executing the command.

The definition for DEMO2 is only a little bit different from DEMO1 , but a glance at Figure 3 shows that completely different results are produced!

The definition is still written to be compiled with BASE set to 16, but, this time, I want all of the output to be in binary. This is most conveniently forced by calling BIN (line 3 of screen #3) before I enter the DO ... LOOP . Remember, this call to BIN is inside the definition, so it has no effect during compilation, only during execution. Therefore, all of the numbers within the definition are still compiled in base-16, and not in base-2.

***** RULE #2: Any change in the base within a definition does not influence compilation; it only influences execution. *****

The call to BIN every time .BIN is executed at every pass through the DO ... LOOP is not really important. It is only redundant and harmless. Redundancy is usually harmless in any operation involving text output, but can be a problem at other times.

```
 214748346  01111111 11111111 11111111 11111110
 214748347  01111111 11111111 11111111 11111111
-214748348  10000000 00000000 00000000 00000000
-214748347  10000000 00000000 00000000 00000001
-214748346  10000000 00000000 00000000 00000010
```

Figure 2. A portion of the output from DEMO1 .

There is a practical reason behind screens #1-3, and I will discuss that another time.

```
 11111111111111111111111111110   01111111 11111111 11111111 11111110
 11111111111111111111111111111   01111111 11111111 11111111 11111111
-10000000000000000000000000000000 10000000 00000000 00000000 00000000
-11111111111111111111111111111   10000000 00000000 00000000 00000001
-11111111111111111111111111110   10000000 00000000 00000000 00000010
```

Figure 3. A portion of the output from DEMO2 .

forth.lst

```
SCR # 1
  0 \ BIN  .BIN                                          \ RDL 12/16/88
  1
  2 FORTH  DEFINITIONS  DECIMAL
  3
  4 : BIN ( - )                                          \ RDL 12/16/88
  5     2  BASE  ! ;
  6
  7 : .BIN  ( - )                                        \ RDL 12/16/88
  8     BIN
  9     <# # # # # # # # # BL. HOLD # # # # # # # # BL HOLD # # # # #
 10        # # # BL HOLD # # # # # # # # #>  TYPE ;

SCR # 2
  0 HEX
  1
  2 : DEMO1 ( - )                                        \ RDL 12/16/88
  3     10  0  DO
  4       CR
  5       I  0    7FFFFFF7.  D+    2DUP
  6       DECIMAL  0C  D.R
  7       2  SPACES
  8       .BIN
  9     LOOP ;
 10
 11 DECIMAL

SCR # 3
  0 HEX
  1
  2 : DEMO2 ( - )                                        \ RDL 12/16/88
  3     BIN
  4     10  0  DO
  5       CR
  6       I  0    7FFFFFF7.  D+    2DUP
  7       21  D.R
  8       2  SPACES
  9       .BIN
 10     LOOP ;
 11
 12 DECIMAL

+++
```

FACET SOFTWARE

ULTRASCIENCE Div., Gibbs Laboratories, Inc.
1824 Wilmette Avenue, Wilmette, IL 60091
Telephone 312/256-0080
Fax 312/256-0097

S/R Software Resource™

The variety of specific applications which can be built around the OS-9 operating system is limitless. The dense kernel and inter linking core modules strike an optimal balance between flexibility and remarkably efficient real-time performance. OS-9 also provides exceptionally powerful interprocess message passing capability, hardware independence, support for essentially all the major high-level languages, ..., and it gets better all the time. What more could you wish for?

Ultrascience provides powerful S/R Facet software, to enhance your interface with OS-9, to facilitate rapid application design, and to extend OS-9 hardware independence for peripheral devices. The list of S/R Facet software is open-ended, and suggestions are always welcome.

Brief Descriptions

S/R TICTOC - is an acronym for "Terminal Input Conversion/Terminal Output Conversion." The TICTOC interface is simpler to use, yet more powerful than termcaps. TICTOC neutralizes differences between terminals, even major differences, so extensively enhanced screens will display properly without programming changes. Using TICTOC, the same program will display correctly and brilliantly on a Wyse 50 (embedded mode), a Link (non-embedded mode), and a VT220 (character mode). TICTOC also converts input from terminals, permitting keyboards to be adapted dynamically to the requirements of an application. The unified, well ordered, standard TICTOC commands provide cursor control, visual enhancements, graphics, function and edit key translations, auxiliary port and printer controls, ... for an ever growing library of "terminal handlers". New handlers are being created all the time, and custom handlers are easily created using the powerful TICTOC MAKER.

S/R SHELL - The Bourne Shell is largely responsible for the current success of UNIX. It is a well established command processing program language, complete with wild carding, variables, pipelines, redirects, tests, structured conditionals, operators, backticks, ... Ultrascience adapted the Bourne Shell for OS-9. Powerful and easily learned, SHELL should be a part of every OS-9 system.

S/R CRON - permits you to set up a list of functions which are to be performed automatically by the system at the time(s) you specify. You may set functions to execute once, or periodically, at a particular time-of-day, day-of-week, or month-of-year. There is no limit to the number of operations you can queue up, and they can be established for as long as a year into the future.

S/R XDIR - provides an interactive graphic display of your file directories. It displays multiple files and directories; you can walk along any directory pathandgetan instant view of the files in the directory. XDIR will optionally display file attributes such as size, owner, date, e/r/w flags, etc. Searches for files based on a wild-card, "regular" expression, can be made recursively, and/or within selected directories. Files and directories can be marked and then used as input to any OS/9 shell command. A VTREE feature allows you to see a graphic display of your directory structure, and a special mode of XDIR makes it behave like the UNIX "find", so that filename matches can be used as standard input in a pipeline command.

S/R MENU - is the quickest way to establish a friendly interface between yourself and the OS/9 system. Menu selections are defined by title, selection key, and the OS/9 system command to be executed; MENU does the rest. It balances your menu display by selection count and title length, draws pretty boxes, inserts menu headers and date/time - you select the style. A single key-stroke is all that is necessary to select an option from a completed menu. Since MENU can execute any OS/9 command, MENU can invoke another MENU; thereby permitting unlimited nesting of menus.

S/R CPIO - Moves data in and out of standard CPIO format for inter-system transfer and tape or floppy storage.

S/R TAR - Moves data in and out of standard TAR format for inter-system transfer and tape or floppy storage.

Hardware Diagnostic Software

Hundreds of gates, thousands of gates, millions of gates..... It is hard enough to find a broken one; but how does one find the intermiuent one, the one which is sensitive to temperature, voltage, or simply has a mind of its own. Board swapping can be a powerful troubleshooting tool, if you have sufficient redundancy in your system or systems; and sophisticated logic probes and bus analyzers can perform wonders. However, a really good diagnosis always makes any repair easier - often trivial enough to complete in the field withouttest equipment. Ultrasciencethought it would be nice to offer a set of do-it-yourself software tools for diagnosing hardware failures; something really easy to use, something that would make it possible to use the sophistication of an OS-9 computer to diagnose itself.

Brief Descriptions

S/R cputest - exercises the 680XX master CPU chip with an extensive battery of Motorola CPU tests (e.g.; addressing modes; arithmetic; data movement; branching instructions; exception processing; and memory management, if applicable; etc.). Pass-fail is reported.

S/R fpctest- challenges the floating point coprocessor with an array of function tests (e.g., move, fsave, frestor; status register reads and writes, exception handling for overflow, etc.), designed to detect malfunction. Pass-fail is reported.

S/R dramtest - tests DRAM with an intense series of challenges, designed to test for complex gate interaction and refresh failures, as well as simple "stuck" bits. The addresses of any errors are reported.

S/R siotest - tests any two serial I/O ports on a system against one another to confirm flow control and data integrity. All 256 ASCII characters are transferred in both directions and errors are reported. Both XON/XOFF and hardware flow control, DSR/DTR or CTS/CTR, are also tested. Errors are reported.

S/R clktest - compares the system clock to the date and time circuit. The tick rate error of the system clock with respect to the date and time circuit is reported. Run in background, clktest will dynamically change the tick rate of the system clock so that the system time will track the time of the date and time circuit. This method of keeping the system time correct is superior to resetting the system clock in a step fashion. In most systems (depending upon the system clock interrupt rate) clktest will maintain the system clock within 5 seconds of the date and time circuit.

S/R tickfix - sets the system time-keeping software's tick rate.

S/R tapetest - records and compares all 256 ASCII characters in a user specified number of tape blocks. Errors are reported.

S/R disktest - performs a high-speed read of all the blocks on a disk. Any replaced blocks and newly detected, defective blocks are reported.

ULTRASCIENCE SAYS THANKS

Thanks to the 68 Micro Journal for being an excellent OS-9 communication port. Thanks to all of you OS-9 folks who gave our PC68K1 coprocessor such a rousing reception when it was announced last month. We knew there was a great deal of interest in placing OS-9 into the PC environment, but we were pleasantly surprised nevertheless. In appreciation for your support, we will make a maximum effort to see that the PC68K1 product remains part of the cutting edge for OS-9 development and mass marketing. A full featured version of OS-9 2.3 is now available for the PC68K1. We have also made a number of enhancements to the PC68K1 specific software. Disk access is now even faster, and high-performance 5 1/4" SCSI disk and tape devices with capacities of as much as 2 gigabytes are supported. The code that dynamically allocates time between DOS and OS-9 has been further optimized, so that persons using DOS concurrently with OS-9 will notice a significant improvement in throughput. Drivers for the twelve OS-9 serial ports have been improved to better handle heavy loads of process control input.

Unfortunately, the PC68K1 has not made everyone happy! So, for those of you who do not now own, nor probably ever will own a PC, we have a present — OS-9 for the MAC. We hope that you will be as pleased as the PC/XT/AT users. Call or write for your specifications. /UNFORMAT

Thank you,

New Product Development
Ultrascience Div., Gibbs Laboratories, Inc.
1824 Wilmette Ave.
P.O. Box 558
Wilmette, Il 60091
TEL: 312/256-0080 FAX: 312/256-0097

+++

FOR THOSE WHO NEED TO KNOW **68 MICRO JOURNAL™**

Minix Runs on the PT68K-2

By:
J. Gary Mills
1019 Weatherdon Ave.
Winnipeg, Manitoba, Canada R3M 2B5

About a year ago, I purchased a PT68K-2 from Peripheral Technology in Marietta, Georgia. I had been looking for a 68000-based microcomputer for some time, and they had come up with one that took advantage of the PC/XT clone components that were widely available. The PT68K-2 is a 68000 board that fits a PC/XT cabinet and has slots for IBM-compatible I/O cards. It has serial and parallel ports and a floppy disk controller, but lacks an MMU or a DMA controller. I ordered one with a megabyte of RAM, an IBM clone keyboard, and an IBM clone monochrome display card.

My PT68K-2 came with the 'Humbug' ROM monitor and the 'SK*DOS/68K' operating system, from Star-K Software Systems. SK*DOS looks like FLEX9, which was familiar to me because I ran it on my 6809 machine. Although SK*DOS is a fine system for many purposes, I was looking for something better. I wanted a unix-like operating system with a hierarchical file system, multi-tasking, and the familiar shell semantics. I had also grown a bit weary of system programming in assembler language. My search led me to consider Minix. The following information is an excerpt from the 'Minix Information Sheet', recently posted to comp.os.minix on USENET.

WHAT IS MINIX?

MINIX is an operating system that is a subset of UNIX Version 7. It contains nearly all the V7 system calls, and these calls are identical to the corresponding V7 calls. It also includes a Bourne-compatible shell, and close to 100 utility programs, including cc, grep, ls, make, etc. To the average user, it is effectively V7 UNIX. If you dig deep enough, you will, however, find some differences.

The MINIX kernel has been written from scratch by Dr. Andrew Tanenbaum <ast@cs.vu.nl>. It does not contain ANY AT&T code at all. The utility programs have been written by Andy Tanenbaum, his students, and a number of other people, including people on USENET. None of the utilities contain any AT&T code either. The shell, the C compiler, make, etc. have all been completely redone. As a result, this code is not covered by the ATT UNIX license, and it can be made available.

What CPUs does Minix run on?

MINIX was originally written for the IBM PC, XT, and AT. It has since been ported to the NS 16032 and the 68000 (Atari ST). It will also work on many 386-based machines.

How can I get Minix?

MINIX is being sold by: Prentice-Hall, Englewood Cliffs, NJ 07632 (1-800-223-1360), and Prentice-Hall Int'l, Hemel, Hempstead, England (+44 442 231555)

When ordering it, please specify one of the following versions:

MINIX for 640K IBM PC $79.95
MINIX for 512K IBM PC/AT $79.95 (0-13-583865-7)
MINIX sources on mag tape $79.95
MINIX code + reference manual (PC) $110 (0-13-584426-6)
MINIX code + reference manual (AT) $110
MINIX for the Atari ST $79.95 (0-13-584392-8)
Textbook: Operating Systems: Design and Implementation (0-13-637406-9)
Reference Manual: MINIX for the IBM PC, XT, and AT (0-13-584400-2)

How Can I Find Our More About Minix?

MINIX is described in detail in the following book:
Title:*Operating Systems: Design and Implementation*
Author:Andrew S. Tanenbaum
Publisher:Prentice-Hall
ISBN: 0-13.637406-9(Hardcover)
0-13-637331-3 (Paperback, outside of U.S and Canada)

A German translation was begun in Feb. 1988. There is also a paperback MINIX Reference Manual that is a subset of the book. It contains only the MINIX specific information, not the general background stuff on operating systems that the book contains The software package does not contain a manual; this is contained in the appendices to the book, which also contain a complete source code listing (in C) of the MINIX kernel.

Is Minix Public Domain?

No. MINIX has been copyrighted by Prentice-Hall. Prentice-Hall has decided to permit a limited amount of copying of the sources and binaries for educational use. Professors may make copies for students in their operating systems classes. Academic researchers may use it for their new experimental machines, and things like that. A small amount of private copying of diskettes for the use of personal friends is ok, but please do not make more than 3 copies from each original. Prentice-Hall is trying to be more reasonable than most software publishers. Please do not abuse this. Online repositories of the full source code distribution are not permitted. All commercial uses of MINIX require written permission from Prentice-Hall; for the most part, they are willing to grant such permission in return for a royalty on sales.

What Comes With Minix?

Minix includes the complete kernel source and binaries. Source for all commands except for the compiler and linker are also included. The following programs come with the Atari ST version:
ar as badblocks basename cal cat cc cdiff chmem chmod chown clr cmp comm compress cpcpdir date dd df diff diskcheck du echo expr factor false find fix fsck getlf grep gres head kill ln login lpr ls make megartc mined mkdir mkfs mknod more mount mv od passwd pr printenv pwd readall readfs rev rm rmdir roff sed sh shar size skdos sleepsort split stty su sum sync tail tar tee test time tos touch tr treecmp true umount uniq update uudecode uuencode wc

ADAPTATION TO THE PT68K-2

The Atari ST has the same 68000 cpu as the PT68K-2, so that version of Minix made a good starting point. However, the peripheral devices are almost entirely different. I borrowed an Atari ST and set to work, doing all the development under Minix, using the Minix compiler. All the kernel source files that were specific to the Atari ST, most of which were device drivers, had file names beginning with 'st'. The first task was to modify these files to suit the peripheral devices on the PT68K-2, creating an equivalent set of files beginning with 'pt'. This took about a month of evenings and weekends, and was made more pleasant by the nicely-written kernel code. Generally, only a small portion of each file needed to be modified. Once that was done, it only remained to build a kernel image and to create a boot disk for the PT68K-2. I won't claim that Minix booted on the PT68K-2 the first time, but it did boot and run the second time, after minor changes. The following section describes the hardware differences and the kernel changes that were required to create a version of Minix that would run on the PT68K-2.

Interrupt Handling

On the Atari, interrupts may be generated by the clock timer, the DMA device, the keyboard ACIA, or the parallel port. Interrupts from hardware are handled by a 68901 multi-function peripheral, which priorizes them and supplies a vector number to the cpu that invokes one of the first sixteen user vectors. Assembler code in the file 'stmpx.s' handles the vectored interrupts and calls interrupt service routines in various device drivers. The PT68K-2 has a more primitive interrupt system, using the non-68000-family interface. Interrupts may be generated by the clock timer, the IBM keyboard, or the parallel printer port. All interrupts from hardware are wired to IRQ5, invoking the level five autovector. For the PT68K-2, the file 'ptmpx.s' handles the one hardware interrupt and must poll status registers in various devices to determine which interrupt service routine to call. This file also reserves storage for 'shadow copies' of some registers in the peripheral devices. The reason for this is some multi-part devices, like the DUART or the PIT, have write-only registers that are shared between parts. Keeping shadow copies allows drivers for each part to be separate and not interfere with each other.

The Clock Timer

The Atari ST has a 2.4576 MHz clock which is divided by a programmable counter in the 68901 MFP to produce interrupts. The interrupt service routine in 'clock.c' does a further division by four to produce the 60 Hz clock tick used by the Minix scheduler and real time clock. Only minor changes were necessary to adapt the clock routines to the PT68K-2. A 3.6864 MHz clock is available to the first 68681 DUART, so that clock is divided by a counter programmed to produce interrupts directly at 60 Hz. No software division is required, resulting in a more efficient kernel. The timer in the 68230 PIT would have been a better choice, but it has no connection to the clock, and no interrupt line. Simple hardware modifications could remedy this. The interrupt should likely be at a higher priority than the keyboard interrupt.

The Keyboard

Changes to the keyboard driver were mainly a result of differences in the keyboard interface because both the Atari keyboard and the IBM clone keyboard transmit the same scan codes. The Atari ST uses a 6850 ACIA whereas the PT68K-2 uses a TTL keyboard register. The PT68K-2 keyboard register interrupts via an input line on the first DUART. Obtaining the scan code requires a read from one address to get the byte, followed by a read from a second address to reset the register. The IBM keyboard has built-in key repeat, so the software repeat routine in the Atari version is no longer needed. The file 'stkbd.c' also contained support for Atari national keyboards. This was deleted as well.

The Display

The Atari ST display is quite different from the IBM PC clone display card used in the PT68K-2. The Atari has a video controller device that uses 16 K of system RAM for a bit map of the screen. The driver copies information from font tables to form characters on the screen. On the PT68K-2, the video RAM and controller are on the display card. Each display position on the screen has a character byte and an attribute byte in video RAM. For the PT68K-2, the driver initializes the video controller registers to start the display with a blank screen. It does scrolling simply by copying bytes in video RAM, and does cursor movement by changing the cursor location registers in the controller. The font tables and associated code, of course, had to be deleted, but all the support for ANSI escape sequences was retained with only minor changes. The display driver also is responsible for the 'bell' tone, and on the Atari, it uses the sound device to generate the tone. On the PT68K-2, sound is produced by enabling and disabling an output from the first DUART that drives the speaker. Unfortunately, the timer in the DUART has to run at 60 Hz to serve as the system clock, but no other timer was available.

The DMA Device

The Atari ST uses a DMA device for access to the floppy disk and the hard disk, managed by routines in the file 'stdma.c'. The PT68K-2 has no DMA, so that data transfers to and from the disks must be done by cpu action. This is a basic limitation of the PT68K-2. The DMA routines are omitted from the PT68K-2 version of Minix, requiring corresponding changes in the floppy disk driver.

The Floppy Disk Driver

The Atari ST uses a Western Digital 1772 floppy disk controller, accessed via the DMA device. In the file 'stfloppy.c', the driver starts each floppy I/O operation by issuing a command to the controller. All operations interrupt on completion, so the interrupt service routine checks the result of the operation and takes appropriate action. The DMA device does the data transfers for sector read and write operations, also interrupting on completion. The floppy driver required considerable modification for the PT68K-2 version because, although the PT68K-2 also uses the WD 1772 FDC, it has no DMA, and the interrupt line is not connected. During sector I/O operations, the data transfer rate is too high to allow the cpu activity to be interrupted by other devices. It is therefore necessary to disable interrupts during these operations. Some interrupts are lost during sector I/O, affecting mainly the clock, but potentially also the keyboard and parallel port. The structure of the driver had to be revised to poll the FDC and wait for completion of each operation. Interrupts are enabled at this point, so other system activity can continue while the floppy driver waits. An attractive hardware modification would be to connect the FDC interrupt line and use interrupts to signal completion. The line should be a low priority interrupt, and would have to pass through a DUART or a PIT so it could be enabled by software when required. One advantage of doing this would be to allow a programmed time out to interrupt the FDC when accessing a drive with no disk inserted.

The Hard Disk Driver

The Atari ST has its own unique hard disk controllers. A driver could have been written to support the Western Digital controller card that the PT68K-2 uses, but the simplest adaptation was to defer this until later. Consequently, the file 'ptwini.c' is only a dummy hard disk driver, based on the Atari version.

The Printer Driver

Changes to the printer driver were mostly due to differences in the hardware. The Atari ST uses a parallel port in the 68901 MFP for a printer port. The PT68K-2 has a printer port on the IBM clone monochrome video card, but it is not usable because it has no interrupt line. However, the parallel port in the 68230 PIT is suitable. Interrupt handling is a bit tricky because the PIT will interrupt whenever the port output buffer is empty. In the file 'ptprint.c', the driver initializes PIT port A for pulsed handshake with interrupts disabled. The driver then only enables the interrupt when output is in progress and more characters remain to be output. This driver has not been tested, but will likely work.

Memory Size Determination

In the file 'mm/main.c', the Atari ST version of Minix reads a TOS variable to determine the memory size. The PT68K-2 version simply assumes that one megabyte of RAM is present. Minix would work with 512 K of RAM, so this could be changed to do a memory test of some sort.

Generic Kernel Files

There were many files under the 'h', 'mm', 'fs', and 'kernel' directories that contained code that is only compiled when the symbol 'ATARI_ST' is defined. These were all enhanced to produce the PT68K-2 version when the symbol 'PT68K' is defined. In many cases, only the symbol was changed, as the Atari code was also appropriate for the PT68K-2.

THE BOOT BLOCK FOR THE PT68K-2

The boot disk for Minix simply consists of a boot loader in the first sector, followed by the kernel image in consecutive sectors. It is conventially on a single-sided diskette. The task of the boot loader is to load the kernel image into memory and start execution. The Atari ST version of Minix used a BIOS call to do the load. For the PT68K-2, the boot block requires routines to drive the WD 1772 floppy disk controller for 'restore', 'seek', and 'read sector' operations. This code fits quite nicely into the 512-byte sector, leaving room for some variables required by Minix. The file 'bootblok.s' is included here as 'Listing 1'. To begin the boot, the Humbug 'fd' command loads the first sector into memory and jumps to the first location. Fortunately, Humbug has no problem loading a 512-byte sector, and the rest is done by the boot loader and the Minix disk driver.

```
! Boot block for the PT68K-2, complete with low level disk i/o
! for the WD1772.  Expects an 80-track single-sided disk in drive 0.
        .sect   .text
        .sect   .rom
        .sect   .data
        .sect   .bss

        .sect   .text
start:
        bra     boot            ! 000: jump to loader
        .ascii  "MINIX "        ! 002: 6 byte identification
        .data1  0,0,0           ! 008: volume serial
        .data1  0,2             ! 00B: 512 bytes/sector (low byte first)
        .data1  2               ! 00D: 2 sectors/cluster
        .data1  1,0             ! 00E: reserved sector (low byte first)
        .data1  2               ! 010: number of FATS
        .data1  112,0           ! 011: number of dirs (low byte first)
        .data1  208,2           ! 013: 720 sectors (low byte first)
        .data1  248             ! 015: media descriptor (80 track SS)
        .data1  5,0             ! 016: sectors/FAT (low byte first)
        .data1  9,0             ! 018: sectors/track (low byte first)
        .data1  1,0             ! 01A: number of sides (low byte first)
        .data1  0,0             ! 01C: hidden sectors (low byte first)

! offsets in this boot block:
magic   = 502
nsect   = 504
fsckd   = 506
zero    = 508
fsckt   = 510

ldaddr          = 0x040000

! disk controller registers
comreg  = 0xFE0101
stareg  = comreg
trkreg  = 0xFE0103
secreg  = 0xFE0105
datreg  = 0xFE0107
dlatch  = 0xFE00C1

boot:
        move.w  #0x0001,d6      ! start with cyl 0, sec 1
        move.w  start+nsect(pc),d4
        move.l  #ldaddr,a3      ! load address in memory

read:
        tst.w   d4
        beq     rel
        bsr     dread
        bne     boot
rel:
        lea     copy(pc),a0
        lea     start(pc),a1
        sub.l   a1,a0
        add.l   #ldaddr,a0
        move.l  a0,0x0014
        divs    #0,d0           ! jump to copy routine in super state

copy:
        move.w  #0x2700,sr
        move.l  #8,a0
        move.l  #ldaddr+0x208,a1        ! start address of minix
```

```
    move.l  #0x400,d0
cp2:  move.l  (a1)+,(a0)+
    cmp.l   a0,d0
    bne     cp2
    add.l   #0x200,a0     ! skip tos variables
    add.l   #0x200,a1
    clr.l   d0
    move.w  start+nsect(pc),d0
    asl.l   #8,d0         ! multiply
    asl.l   #1,d0         ! with 512
cp3:  move.l  (a1)+,(a0)+
    cmp.l   a0,d0
    bne     cp3
    move.l  ldaddr+0x204,a0
    jmp     (a0)          ! minix boot adres


dread:
    move.b  #0x20,dlatch  !side 0, dd, drive 0
    bra     dr2           !goto restore
dr1:
    bsr     sread         !read sector
    beq     dr3           !until successful
    add.w   #1,d3         !incr error count
    cmp.w   #10,d3        !until too many errors
    blt     dr1           !loop
dr2:
    move.b  #01,comreg    !restore
    bsr     wnbusy        !wait for completion
    clr.w   d3            !no errors now
    bra     dr1           !loop
dr3:
    add.w   #512,a3       !incr load addr
    add.b   #1,d6         !incr sector
    cmp.b   #9,d6
    ble     dr4           !if past cyl
    clr.b   d6            !reset sector
    add.w   #0x0101,d6    !calc next cyl
dr4:
    sub.w   #1,d4         !decr count
    bgt     dr1           !until all done
    rts                   !return

sread:
    move.l  a3,a2         !-->place for data
    move.w  d6,d7         !get next track sector
    move.b  d7,secreg     !give sector to fdc
    asr.w   #8,d7         !get track
    cmp.b   trkreg,d7     !if different track
    beq     sr1
    move.b  d7,datreg     !give track to fdc
    move.b  #0x11,comreg  !seek
    bsr     wnbusy        !wait for completion
sr1:
    lea     datreg,a0     !-->data reg
    lea     stareg,a1     !-->status reg
    move.b  #0x84,comreg  !read
    bsr     wait

sr2:
    move.b  (a1),d0       !check status
    btst    #1,d0         !drq?
    bne     sr3
    btst    #0,d0         !busy?
    bne     sr2
    bsr     wnbusy        !wait for completion
    and.b   #0x1C,d0      !mask errors
    rts                   !return

sr3:
    move.b  (a0),(a2)+    !get a byte
    bra     sr2           !loop

wait:
    clr.b   d7
wa1:
    sub.b   #1,d7
    bne     wa1
    rts

wnbusy:
    bsr     wait
    move.b  stareg,d0     !get status
    btst    #0,d0         !busy?
    bne     wnbusy        !loop
    rts                   !return with status
```

REQUIREMENTS FOR PT68K-2 MINIX

To run Minix on a PT68K-2, you need one megabyte of RAM, an IBM clone keyboard and monochrome display card, and at least one 80-track double-sided 3.5" floppy disk drive. It's not possible to use a terminal as the console because neither the Atari version nor the PT68K-2 version includes a serial port driver. You also, of course, need the Atari version of Minix, which comes with nine 3.5" diskettes and a 62-page manual. All diskettes except the 'boot' and 'tos' diskettes are usable on the PT68K-2.

RESULTS

Minix runs beautifully on the PT68K-2 - in some ways, better than on the Atari ST. It does, however, have limitations, and certain enhancements will likely require hardware modifications to the PT68K-2. It definitely feels like Unix. It's very solid. There are a few bugs, many of them reported on USENET, but just about everything works well, and works as expected. Having the source code for the kernel and the commands is a great advantage. When bugs are reported, and patches posted, it's very easy to apply updates and build a new binary. The emacs-inspired screen editor and the C-compiler work very nicely. Finally, because Minix is compatible with Unix, there are all those public-domain Unix source programs available, most of which will run on Minix with little or no modification. A programmer will feel right at home in this environment.

FOR THOSE WHO NEED TO KNOW **68 MICRO JOURNAL™**

Logically Speaking

The Mathematical Design of Digital Control Circuits

Most of you will remember Bob from his series of letters on XBASIC. If you like it or want more, let Bob or us know. We want to give you - *what you want!*

By: R. Jones
Micronics Research Corp.
33383 Lynn Ave., Abbotsford, B.C.
Canada V2S 1E2
Copyrighted © by R. Jones & CPI

SOLUTIONS TO TEST SIXTEEN-A

And now let's round off with the continuation of our yesterday-morning's bed-time story!

1.

X_1X_2 00	01	11	10	Z
①	2	3	4	0
1	②	5	6	1
1	7	③	6	0
1	2	5	④	0
8	2	⑤	4	1
8	7	5	⑥	1
8	⑦	3	6	0
⑧	2	5	6	1

No combinations are possible.

which he would insert the reply he'd LIKE to get.

Then he completed the various rows. As an example, row 1 indicates that if the info-guy were a Bingo, and the chosen fork actually led to the Lost City, he would reply YES, which would also be the desired response. No problem at all with Bingos, but those darned Bongos .... Unlike Uncle Fred's situation, where his K-map defined a statement to be made, Joe was looking for a question to be asked, so he tried reading out the headings, to the left of his desired response, as a crude form of question. This is what he did, and addressing himself to the info-man, at the same time pointing down one of the forks, he asked

MAN	ROAD	REPLY	DESIRED
1	1	1	1
1	0	0	0
0	1	0	1
0	0	1	0

"IF I were to ask you if this is the road to the Lost City, what would you reply?"

Alternatively, instead of posing a general question of this type, he COULD have selected, say, the first row of his truth-table, and asked a more specific question, such as

"IF I were to ask you if this is the road to the Lost City, would you say YES?"

Note the emphasis on the first "IF". In other words, Joe's not REALLY asking the question - he only wants to know what the man would say IF he were to ask it! Of course, if the info-man were a Bingo and the road did in fact lead to the Lost City, his reply (IF he were asked) would be YES, and (being

THE LOST CITY AND KING SOLOMON'S TREASURE

Joe remembered that he HAD to get to the Lost City, so he decided to draw up a truth-table instead of a K-map, to set out what would happen if he selected one of the forks at random, pointed down it, and asked the info-man if this were the road to the Lost City. To start with, he defined the variables of the situation as follows

Bingo = 1	Road to City = 1	Reply - YES = 1
Bongo = 0	to Cannibals = 0	NO = 0

and then he drew up a truth-table with the first column headed MAN, the second ROAD ACTUALLY LEADS TO, which he shortened to ROAD, and the third column with the REPLY he'd get from the info-guy. Finally, in order to cover as much data as possible, he added a fourth column in

2.

(i)

X_1X_2 00	01	10	Z_1Z_2
①	2	3	01
1	②	-	00
4	-	③	00
④	2	3	10

(ii)

X_1X_2 00	01	10	Z_1Z_2
①	2	3	11
4	②	-	00
5	-	③	00
④	2	3	01
⑤	2	3	10

(iii)

X_1X_2 00	01	10	Z_1Z_2
①	2	3	10
4	②	-	00
1	-	③	00
④	2	3	01

a truth-teller) he would admit to this fact and reply YES to the first question. Similarly to the second. On the other hand, if he were a Bongo and the road did in fact lead to the Lost City, his reply (IF he were asked) would be NO, but (being a liar) he's going to lie about this and tell Joe that he'd say YES. In either case, Bingo or Bongo, Joe would get the correct reply, still without knowing to which tribe the guy actually belonged!

I leave it to you to figure out that if the fork selected did NOT lead to the Lost City, Joe would again get the desired answer of NO.

A third possible question would be based on getting a Bingo to tell the truth about a lie (and conversely a Bongo to tell a lie about the truth) with a question of this sort

"If you belonged to the other tribe, and I were to ask you if this is the road to the Lost City, what would you say?"

In this case, Joe would have to complement the reply in order to get the right response.

And so Joe reached the Lost City after all, filled up his pockets with as many jewels as he could carry, and returned home a wealthy man. Then he married a beautiful lady, who bore him several children. Needless to say, one of the first things he did was to draw up a new map to hand down to HIS eldest son when he came of age. And, of course, he made sure that ALL his children were well-schooled in Boolean Algebra, truth-tables, etc., and how to manipulate 1s and 0s in strange situations. So they all lived happily ever after!

Wasn't that a terrific story? Happy ending and all!! In return for all this useful knowledge, maybe someone out there can help ME with a problem. Years ago, I used to collect all these logic-problems, but along the way I seem to have mislaid another, much more complex one dealing with Bingos and Bongos. This one had a third tribe, called Bungos, their peculiarity being that each successive statement they made alternated between being the truth and being a lie. Each Bungo, at sunrise, would individually and randomly decide whether his first statement of the day would be the truth or a lie, and from then on, for the rest of the day, he'd alternate. So you never knew where you stood with a Bungo! I can't for the life of me remember what the logic-problem was, so if ANYONE has ever come across it, I'd appreciate your letting me know! Thanks!!

Now back to serious business!

Mile 22 - heading for Mile 23

RANDOM-INPUT SEQUENTIAL CIRCUITS (continued)

MERGING THE COMBINED FLOW-TABLE

Combining, you'll remember, consists of eliminating redundant STABLE-STATES, still leaving only one stable state per row. Merging, on the other hand, consists of eliminating redundant ROWS by placing as many stable states as possible in one row. Commencing with a primitive flow-table, we "combine" first, then we "merge". Don't get the two operations mixed up!

The rules for merging are the same as for a normal sequential flow-table, EXCEPT THAT THE OUTPUTS ARE IGNORED ALTOGETHER, as they do not come into the picture at this stage. I mentioned earlier that the output shown in the Z-column belongs only to the circled stable state in its row, and as we're concerned with merging ROWS and not combining STABLE-STATES, the question of compatible outputs does not arise. Quite unlike normal sequential flow-tables where we needed to take care that the Sections-C were compatible!!

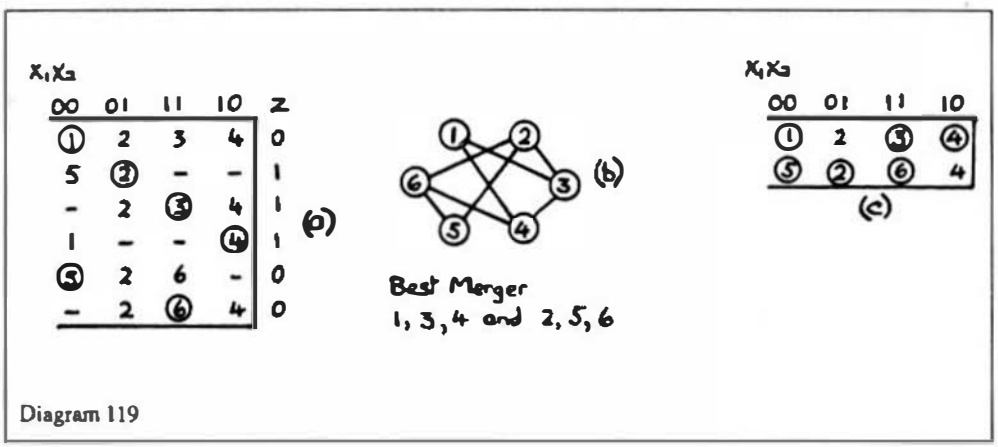


Diagram 119

When two or more rows are merged, all stable states remain circled. For example, the merger diagram of Diagram 119b shows that rows 1, 3 and 4 of 119a can be merged, and so in the merged flow-table of 119c states 1, 3 and 4 are circled in row 1. Similarly, 2, 5 and 6 are merged into row 2 of the merged flow-table, with states 2, 5 and 6 circled. Note that no output conditions appear in the merged flow-table.

CONVERTING THE MERGED FLOW-TABLE TO STANDARD FORM

The merged flow-table of Diagram 119c has two rows and four columns, so a 2-row, 4-column standard flow-table is drawn up as shown in Diagram 120a, with the stable states of row-1 of 119c converted to stable states in the standard table, by entering 1s in Section-A of columns 00, 11 and 10. In the same manner, the stable states of row-2 of 119c now correspond to an entry of "2" in the appropriate Sections-A of the standard table. The unstable state designations are self-evident!

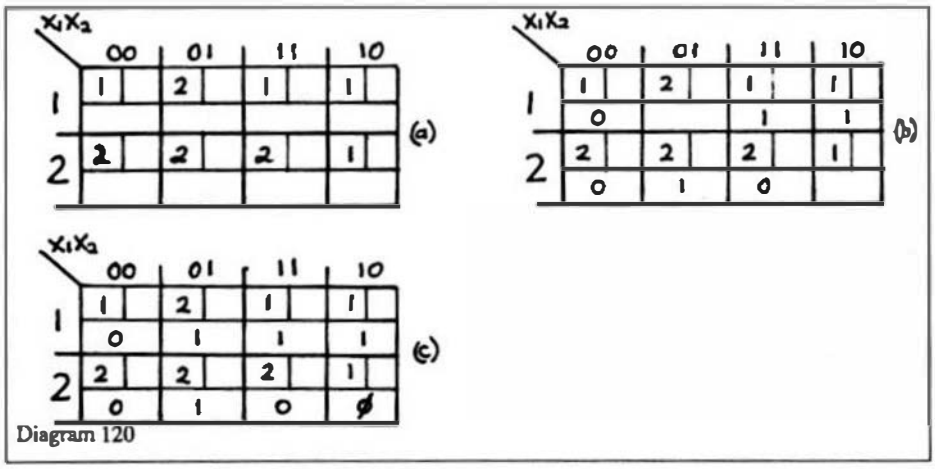


Diagram 120

INCORPORATING THE OUTPUT CONDITIONS IN THE STANDARD TABLE

Having got that far, we now record the output entries in Section-C FOR THE STABLE STATES ONLY, as illustrated in Diagram 120b, the information being obtained from the primitive flow-table

Completing the UNstable states is a little trickier, and is done as follows. Commencing with UNstable-state-2 in row-1 of 120b, we transfer our attention to the corresponding entry in 119c, where we find that this is also an unstable-state-2, signalling a move to the stable-state-2 below. Such a move can only be made from one or other of the stable states in row-1, ie, 1, 3 or 4, but a study of the corresponding conditions in the primitive flow-table of 119a shows that in fact it's possible to make this move only from stable-state 1 or 3, the only move from stable-state-4 being into 1.

Returning now to the blank Section-C in row-1 of 120b, we've established that we can cycle through this address only from one or other of the squares directly adjacent to it. Because address 00.1 has an output of 0 and address 11.1 an output of 1, ending up in row-2 with a 1, we MUST insert a 1 in our blank square. Let's examine the reasons behind this decision for a moment.

Assuming we inserted a phi (which we'd normally do on an elbow), it's possible that in the interests of optimum decoding we'd elect to read this phi as a 0. In that case, commencing from address 00.1 we'd have an output of 0, which would remain 0 in 01.1 and switch to a 1 when the action arrived at 01.2. So far so good! But supposing we commenced at address 11.1 with an output of 1. This would change to a 0 in address 01.1, then back to a 1 again when the action settled in 01.2. Such a switch from ON to a momentary OFF and back ON again is a most undesirable state of affairs, producing a "glitch", which is to be avoided at all costs.

Of course, we may choose to interpret the phi as a 1, and then what would happen? Starting at address 00.1 with a 0-output, we'd switch to a 1, then down to another 1 - - quite a smooth change. If we started with the 1-output, it would remain a 1 throughout the whole operation. So, to avoid the possibility of reading the phi as a 0, with its resultant glitch (even though the circuitry might be simpler) we must make a definite decision NOW to make it a 1, and avoid this problem altogether.

The rule is that where two complementary outputs change across an elbow to one definite output, the output which would remain UNCHANGED has priority in the transition-address, in order to avoid glitches.

When we come to look at unstable-state-1 in row-2 of 120b, the corresponding condition in 119c seems to indicate a possibility of moving out of any of the stable-states 5, 2 or 6, but a check of the primitive flow-table shows that such a move is allowed only from stable-state-6. Back in 120b then, this means that a move is only possible from address 11.2 with 0-output, and so we can insert a phi in the normal way, as there is no conflict of interest with competing output conditions.

The completed standard flow-table is shown in 120c, and it now remains only to Gray-Code a state-diagram, and then code and decode the flow-table in order to obtain the final circuit.

Note that until the standard flow-table is fully completed, we still have to make extensive use of the primitive flow-table in order to determine the output conditions for the UNSTABLE states.

This being the end of random-input sequential circuit design, let's try our hand at

TEST SIXTEEN-B

1. Merge the following combined flow-table, and convert to standard form.

2. Design a circuit having the specifications of the primitive flow-table of Diagram 112, with the following amendments

(i) All input changes are possible, and
(ii) If X1 changes state simultaneously with X2 turning ON, the output-state is optional.

X_1X_2 00	01	11	10	Z
(1)	2	3	–	0
1	(2)	–	4	1
–	–	(3)	5	0
1	2	–	(4)	1
6	2	3	(5)	0
(6)	7	8	–	1
6	(7)	–	5	0
6	7	(8)	–	1

3. Design a circuit with two inputs, X1 and X2, and one output, Z. Z is to be ON only (i) when X1.X2 = 01, OR (ii) when X1.X2 = 11 immediately following 01. If X1.X2 = 11 immediately following 00, Z will be optional. All input changes are possible.

Now for something you're really going to enjoy! We'll take a look next at error-detecting and error-correcting codes, and how they work, which should be quite a pleasant experience.

ERROR-DETECTING AND ERROR-CORRECTING CODES

OTHER TYPES OF CODES AND THEIR PROPERTIES

At various moments during our journey, we've made use of two types of codes, the straight binary and the Gray-code, or reflected binary code. You'll see why it's called that in a little while. However, before going on to discuss the principles of error-detecting codes, we'll first take a look at a few other types of code.

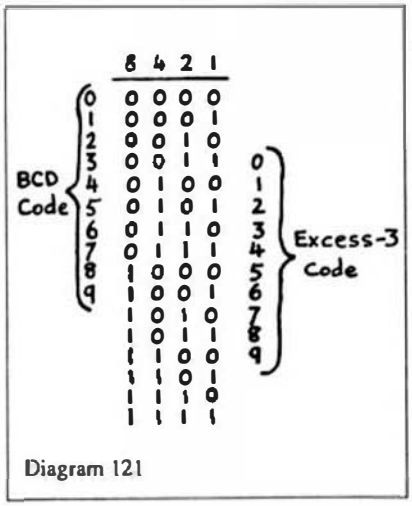


Diagram 121

The center four columns of Diagram 121 show a normal (or straight) 4-bit binary code. To the left is marked off the Binary-Coded-Decimal, or BCD, code, which uses the first ten characters of the straight binary code, while to the right is shown the excess-3 code, which uses the middle ten characters in the straight binary code.

The decimal number 68, which is symbolized by (68)10, would normally be represented by 1000100, symbolized by $(1000100)_2$. In the BCD code EACH INDIVIDUAL DIGIT is represented by a 4-bit character, so $(68)_{10}$ would appear as 0110 1000, while in excess-3 it would appear as 1001 1011, each 4-bit character representing the corresponding decimal digit PLUS THREE.

A very useful property of the excess-3 code is that the 9's complement of a decimal digit can be obtained by complementing all bits. The 9's complement of a number is the difference between that number and 9. Thus the 9's complement of 3 (that is, 6) is obtained by complementing all the bits of 0110 to give 1001. This property makes it possible to "count-down" on a count-up-only counter. For example, suppose a single decade of such a counter had "counted up" to the figure 7 (1010) from some source of pulses, and we now desire to count down by exactly the same number. Instead of designing a reversible counter, which involves quite a bit of extra circuitry, we merely complement our counter to give the 9's complement of 7, (that is, 2, or 0101), and then carry on counting-up in the normal way. When an output is obtained from the TWO MOST-SIGNIFICANT bit-positions (that is, the 8-bit and the 4-bit), we know that a count-down of 7 has been achieved. You'll notice, in the excess-3 code, that whenever a number is complemented, it's now as far from one end of its sequence as it was from the other before complementation took place.

These two codes are based on a block of ten characters taken from the straight binary code. If similar groupings are taken from the reflected, or Gray, code, we obtain the reflected BCD code and the reflected excess-3 code (see Diagram 122). The latter code has the enormous advantage that the 9's complement can be obtained by complementing only the highest-order bit, instead of all bit-positions.

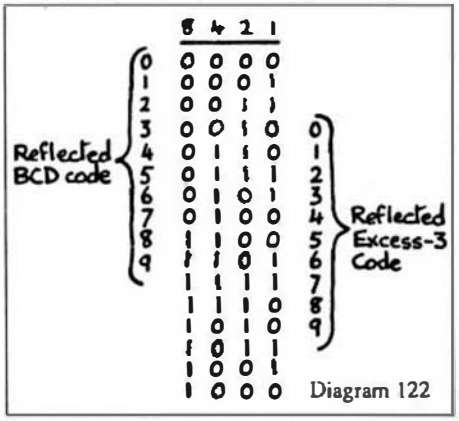


Diagram 122

Now you can see why this particular gray-code sequence is also called a reflected binary code! All characters, except for the most-significant bit, are reflected about the mid-point of the table. Only in the MS-bit position is a column of 0s reflected as a column of 1s. If we wished to extend this table, all we'd have to do would be to "reflect" the current table, and add a new bit-position to the left, with the upper-half all 0s and the lower-half all 1s. And so on, and so on.

MINIMUM DISTANCE OF A CODE

The "minimum distance" of a code is equal to the number of bits which must be changed in a character to produce another valid character. That is to say that if you select any character in the code system of, say, Diagram 122, and change any ONE of its bits, the resulting character will be found in the system. In all codes discussed so far, therefore, the minimum distance is 1.

The "distance" between two coded characters is the number of bits which must be changed in one so that the other results. For instance, the "distance" between 0011 and 0100 is three, since three bits must be changed to transform 0011 into 0100, or vice-versa.

ERROR-DETECTION AND ERROR-CORRECTION

No codes with a minimum distance of 1 can be error-checked, because an erroneous signal in any one bit-position would result in another valid character, and our device would have no way of "knowing" that an error had occurred.

An error-detecting code is defined according to the maximum number of errors it will ALWAYS detect. Thus, if the code can detect ALL single and double errors, and SOME triple (or greater) errors, it is known as a double-error detecting code. An error-correcting code is similarly defined according to the maximum number of errors it will ALWAYS correct. Before it can CORRECT an error, an error-correcting code obviously has to DETECT the error first.
Codes with a minimum distance of two can be used for single-error detection, as a single-bit failure in any bit-position will produce a character which will not match any other valid character in the table. A 2-bit failure, however, MAY match another character and the circuit will therefore behave as though there were no error at all, and would carry out the erroneous instruction contained in the supposedly valid code.

A minimum-distance-3 code is capable of detecting double-errors, but if used for CORRECTION purposes it's limited to single-error correction. The key to error-correction is that it MUST be possible to identify the bit in error! A single-bit failure in a minimum-distance-3 code will not match exactly any character in the table, BUT IT WILL COME WITHIN ONE BIT OF MATCHING THE CORRECT CHARACTER. The bit which does not match is then changed to bring about the correction.

A simple minimum-distance-2 code used for error-detection is the odd-parity code. This is developed by adding an extra "parity-check" column to the particular coding being used, either to the left or to the right of the regular code. In the example shown in Diagram 123, the basic code is a 4-bit Gray-code in the columns y1, y2, y3 and y4, and then a 1 or a 0 is inserted in each row's parity-bit to make an ODD number of the total number of 1s appearing in that row. The appearance of an EVEN number of 1s during the cycling of a device coded in this way would then indicate an error, though it would not be possible to identify the actual error. It might be just the parity-bit itself, and the actual code-part be OK! For example, the appearance of the character 00110 would indicate a single-bit failure, which COULD have occurred in any of the first three characters of the table. Double errors would, of course, not be detected, as the parity would check out as ODD, and therefore OK.

y1	y2	y3	y4	P
0	0	0	1	0
0	0	1	1	1
0	1	1	1	0
1	1	1	1	1
1	1	1	0	0
1	1	0	0	1
1	0	0	0	0
1	0	0	1	1

Diagram 123

Sometimes even-parity codes are used, but are not so popular, as a total power-failure at the "sending" end would result in an all-0 state (an even number) and the circuit at the "receiving" end would revert to this state, assuming it to be a valid character.

FIXED-BIT CODES

Another class of code is the "fixed-bit", or m-out-of-n code, in which there's a fixed number of 1s per character, such as the 2-out-of-5 code, where all the 5-bit characters contain exactly two 1s. These codes can detect ALL single errors, as the number of 1s will then be either one too many or one too few. They will also detect double-errors involving two 1s or two 0s, but not, of course, double-errors in which a 1 becomes a 0 AND a 0 becomes a 1. All triple errors can be detected.

THE HAMMING CODE - SINGLE ERROR CORRECTION

The Hamming code is rather strange to develop, as the check-bits (note the plural) are inserted right in amongst the normal coding of a character. In order to determine how many bit-positions are required to transmit information in a Hamming code, the bit-positions are first numbered sequentially from left to right as 1, 2, 3, 4, 5, etc., and the bit-positions corresponding to the normal binary powers, ie, 1, 2, 4, 8, 16 ... are reserved for check-bits. The remaining positions only are used for transmitting the actual code pattern.

1	2	3	4	5	6	7	
C1	C2	8	C3	4	2	1	
		1		0	1	1	(a)
0		1		0	1	1	(b)
0	1	1		0	1	1	(c)
0	1	1	0	0	1	1	(d)

Diagram 124

Suppose we wish to transmit the binary number 1011 (decimal 11). We'd first write the header-number 1 (reserving this position for check-bit C1), then 2 (reserving this for check-bit C2), then 3, which we'd use for the high-order bit of our desired number, that is, the 8-bit. The next number, 4, would be reserved for check-bit C3, after which the next three numbers (5, 6 and 7) would be used up for the remaining bits of our basic code. Sounds complicated, doesn't it, but wait till you see what this code can do!!!

Now we enter our desired number, 1011 (see row-a of Diagram 124, and C1 is then allocated a 1 or a 0 to establish EVEN parity over columns 1, 3, 5 and 7 only. An easy way to grasp the idea behind this code is to say to yourself "C1 means that, commencing on C1 itself, I must look at one bit, miss one bit, look at one, miss one, etc., keeping count of 1s all along the row in those bits I look at". If the number is EVEN, enter a 0 under C1, otherwise enter a 1. See row-b for this.

Next, a 1 or a 0 is allocated to C2 to establish EVEN parity over bit-positions 2, 3, 6, 7, as shown in row-c. In other words, commencing on C2 itself, we look at two bits, skip two bits, look at the next two, skip two, and so on, again keeping count of 1-bits along the way.

Finally C3 is chosen to establish EVEN parity over bit-positions 4, 5, 6 and 7, as in row-d, which means looking at four bits (commencing with C3), skipping four, and so on. The number in the header above thecheck-bit concerned tells you how many to take into account, and to skip. The final coded character we're going to transmit now reads 0110011, the true number being in bit-positions 3, 5, 6 and 7.

In order to see how this code works, let's introduce an error in bit-position-5, and transmit the erroneous character 0110111. How can we identify what's gone wrong when this code turns up at the receiving end? It's done by applying the three parity-checks, one at a time, to the character received. If the check shows EVEN parity (that is, correct parity) a 0 is recorded, but if it shows ODD parity (incorrect parity) a 1 is recorded. The resulting binary number, when converted to decimal, indicates the position in error.

Let's do just that, shall we? C1 indicates ODD parity over bit-positions 1, 3, 5 and 7, so a 1 is recorded. C2 indicates EVEN parity over bit-positions 2, 3, 6 and 7 and so a 0 is recorded to the left of our previous record, giving 01 so far. Finally C3 indicates ODD parity over bit-positions 4, 5, 6 and 7, and so a 1 is recorded in front of the 01, resulting in 101, which, of course, is equal to the decimal number 5. This, as we already know, IS the bit-position in which the error has occurred, and to correct it, the bit in position-5 is complemented. If the result of the check shows 000, then all is in order, and no error has occurred.

Only single errors can be detected and corrected with this code. Double errors will appear as a single error and a false correction will be made. These codes are intended to serve only as examples. There are several other codes, and also methods other than coding which permit us to carry out error-detection and correction, but these won't be gone into here, as this leg of our journey is intended merely to introduce you to the subject, and to give you some insight into the principles behind these codes.

And guess what? You're right! Another TEST!!! But a relatively easy one!

TEST SEVENTEEN

Develop the Hanuning code characters for the following decimal numbers, introduce an error into the bit-position indicated, and then apply the parity checks to verify the fact that the code does function correctly FOR SINGLE ERRORS ONLY.

1. Decimal 9 Error in bit-position 6
2. " 8 " " " 7
3. " 10 " " " C2
4. " 7 " " " C3
5. " 7 " " " 3 and 5

OUR USUAL CHIT-CHAT

Next time, it's back to some up-hill travel once more with Boolean matrices in the SYNTHESIS of bridge and non-planar networks. So enjoy yourselves while you can!!

... End of Mile 22, with everyone relaxing (?) at marker Mile-23.

+++

Telephone: (615) 842-4600

South East Media

OS-9, UniFLEX, FLEX, SK*DOS SOFTWARE

Fax: (615) 842-7990

ASSEMBLERS

ASTRUK09 from S.E. Media -- A "Structured Assembler for the 6809" which requires the TSC Macro Assembler.
FLEX, SK-DOS, CCF - $99.95

Macro Assembler for TSC - The FLEX, SK-DOS STANDARD Assembler.
Special -- CCF $35.00; FLEX, SK-DOS $50.00

OSM Extended 6809 Macro Assembler from Lloyd I/O. -- Provides local labels, Motorola S-records, and Intel Hex records; XREF. Generate OS-9 Memory modules under FLEX, SK-DOS.
FLEX, SK-DOS, CCF, OS-9 $99.00

Relocating Assembler/Linking Loader from TSC. -- Use with many of the C and Pascal Compilers.
FLEX, SK-DOS, CCF $150.00

MACE, by Graham Trott from Windrush Micro Systems -- Co-Resident Editor and Assembler; fast interactive A.L. Programming for small to medium-sized Programs.
FLEX, SK-DOS, CCF - $75.00

XMACE -- MACE w/Cross Assembler for 6800/1/2/3/8
FLEX, SK-DOS, CCF - $98.00

DISASSEMBLERS

SUPER SLEUTH from Computer Systems Consultants Interactive Disassembler; extremely *POWERFUL!* Disk File Binary/ASCII Examine/Change, Absolute or FULL Disassembly. XREF Generator, Label "Name Changer", and Files of "Standard Label Names" for different Operating Systems.
Color Computer SS-50 Bus (all w/ A.L. Source)
CCD (32K Req'd) Object Only $49.00
FLEX, SK-DOS $99.00 - CCF Object Only $50.00 UniFLEX $100.00
CCF, with Source $99.00 OS-9, $101.00 - CCO, Object Only $50.00
68010 SUPER SLEUTH - Similiar to 8-Bit Version except written in "C".
68010 Disassembler $100.00 FLEX, UniFLEX, UNIX, XENIX, MS-DOS, SK-DOS, OS-9
OS-9/68K Object Only $100.00 or with Source $200.00

DYNAMITE+ -- Excellent standard "Batch Mode" Disassembler. Includes XREF Generator and "Standard Label" Files. Special OS-9 options with OS-9 Version.
CCF, Object Only $100.00 - CCO, Object Only$ 59.95
FLEX, SK-DOS , Object Only $100.00 - OS-9, Object Only$150.00
UniFLEX Object Only $300.00

CROSS ASSEMBLERS

CROSS ASSEMBLERS from Computer System Consultants -- Supports 1802/5, Z-80, 6800/1/2/3/8/11/HC11, 6804, 6805/HC05/ 146805, 6809/ 00/01, 6502 family, 8080/5, 8020/1/2/35/C35/39/ 40/48/C48/49/C49/50/ 8748/49, 8031/51/8751,32000 and 68000/68010 Systems. Assembler and Listing formats same as target CPU's format. Produces machine independent Motorola S-Text. Includes Macro Pre-Processor. Written in "C". *68000 or 6809 *Macintosh,*Atari, FLEX, CCF, UniFLEX, OS-9, XENIX, UNIX, MS-DOS, SK-DOS*
any object $50 or any 3 for $100
any source is an additional $50 or any 3 for $100
Set of ALL object $200.00 - with source $500.00

XASM Cross Assemblers for FLEX, SK-DOS from S.E. MEDIA -- This set of 6800/1/2/3/5/8, 6301, 6502, 8080/5, and Z80 Cross Assemblers uses the familiar TSC Macro Assembler Command Line and Source Code format, Assembler options, etc., in providing code for target CPU's.
Complete set, FLEX, SK-DOS only - $150.00

CRASMB from LLOYD I/O -- Supports Motorola's, Intel's, Zilog's, and other's CPU syntax for these 8-Bit microprocessors: 6800, 6801, 6303, 6804, 6805, 6809, 6811 (all varieties); 6502, 1802/5, 8048 family, 8051 family, 8080/85, Z8, Z80, and TMS-7000 family. Has MACROS, Local Labels, Label X-REF, Label Length to 30 Chars. Object code formats: Motorola S-Records (text), Intel HEX-Records (text), OS-9 (binary), and FLEX, SK-DOS (binary). Written in Assembler ... e.g. Very Fast.

CPU TYPE - Price each:

For:	MOTOROLA	INTEL	OTHER	COMPLETE SET
FLEX9	$150	$150	$150	$399
SK-DOS	$150	$150	$150	$399
OS-9/6809	$150	$150	$150	$399
OS-9/68K	------	------	------	$432

CRASMB 16.32 from LLOYD I/O -- Supports Motorola's 68000, and has same features as the 8 bit version. OS9/68K Object code Format allows this cross assembler to be used in developing your programs for OS-9/68K on your OS-9/6809 computer.
FLEX, SK-DOS, CCF, OS-9/6809 $249.00

COMMUNICATIONS

CMODEM Telecommunications Program from Computer Systems Consultants, Inc. -- Menu-Driven; supports Dumb-Terminal Mode, Upload and Download in non-protocol mode, and the CP/M "Modem7" Christensen protocol mode to enable communication capabilities for almost any requirement. Written in "C".
FLEX, SK-DOS, CCF, OS-9, UniFLEX, UNIX, XENIX, MS-DOS, with Source $100.00 - without Source $50.00

X-TALK from S.E. Media - X-TALK consists of two disks and a special cable, the hookup enables a 6809 SWTPC computer to dump UniFLEX files directly to the UniFLEX MUSTANG-020. This is the ONLY currently available method to transfer SWTPC 6809 UniFLEX files to a 68000 UniFLEX system. Gimix 6809 users may dump a 6809 UniFLEX file to a 6809 UniFLEX five inch disk and it is readable by the MUSTANG-020. The cable is specially prepared with internal connections to match the non-standard SWTPC SO/9 I/O Db25 connectors. A special SWTPC S+ cable set is also available. Users should specify which SWTPC system he/she wishes to communicate with the MUSTANG-020. The X-TALK software is furnished on two disks. One eight inch disk contains S.E. Media modem program C-MODEM (6809) and the other disk is a MUSTANG-020 five inch disk with C-MODEM (68020). Text and binary files may be directly transferred between the two systems. The C-MODEM programs are unaltered and perform as excellent modem programs also. X-TALK can be purchased with or without the special cables, but this special price is available to registered MUSTANG-020 users only.
X-TALK Complete (cable, 2 disks) $99.95
X-TALK Software (2 disks only) $69.95
X-TALK with CMODEM Source $149.95

XDATA from S.E. Media - A COMMUNICATION Package for the UniFLEX Operating System. Use with CP/M, Main Frames, other UniFLEX Systems, etc. Verifies Transmission using checksum or CRC; Re-Transmits bad blocks, etc.
UniFLEX - $299.99

Availability Legends
O = OS-9, S = SK*DOS
F = FLEX, U = UniFLEX
CCO = Color Computer OS-9
CCF = Color Computer FLEX

VISA

South East Media

5900 Cassandra Smith Rd. - Hixson, Tn. 37343

Telephone: (615) 842-4600 FAX (615) 842-7990

master card

** Shipping **
Add 2% U.S.A. (min. $2.50)
Foreign Surface Add 5%
Foreign Airmail Add 10%
Or C.O.D. Shipping Only

*OS-9 is a Trademark of Microware and Motorola-*FLEX and UniFLEX are Trademarks of Technical Systems Consultants-*SK*DOS is a Trademark of Star-K Software Systems Corp.

Telephone: (615) 842-4600

South East Media

OS-9, UniFLEX, FLEX, SK*DOS SOFTWARE

Fax: (615) 842-7990

PROGRAMMING LANGUAGES

PL/9 from Windrush Micro Systems -- By Graham Trott. A combination Editor Compiler Debugger. Direct source-to-object compilation delivering fast, compact, re-entrant, ROM-able, PIC. 8 & 16-bit Integers & 6-digit Real numbers for all real-world problems. Direct control over ALL System resources, including interrupts. Comprehensive library support; simple Machine Code interface; step-by-step tracer for instant debugging. 500+ page Manual with tutorial guide.

FLEX, SK-DOS, CCF - $198.00

PASC from S.E. Media - A FLEX9, SK-DOS Compiler with a definite Pascal "flavor". Anyone with a bit of Pascal experience should be able to begin using PASC to good effect in short order. The PASC package comes complete with three sample programs: ED (a syntax or structure editor), EDITOR (a simple, public domain, screen editor) and CHESS (a simple chess program). The PASC package comes complete with source (written in PASC) and documentation.

FLEX, SK-DOS $95.00

WHIMSICAL from S.E. MEDIA Now supports *Real Numbers*. "Structured Programming" WITHOUT losing the Speed and Control of Assembly Language! Single-pass Compiler features unified, user-defined I/O; produces ROMable Code; Procedures and Modules (including pre-compiled Modules); many "Types" up to 32 bit Integers, 6-digit Real Numbers, unlimited sized Arrays (vectors only); Interrupt handling; long Variable Names; Variable Initialization; Include directive; Conditional compiling; direct Code insertion; control of the Stack Pointer; etc. Run-Time subroutines inserted as called during compilation. *Normally produces 10% less code than PL/9.*

FLEX, SK-DOS and CCF - $195.00

KANSAS CITY BASIC from S.E. Media - *Basic for Color Computer OS-9* with many new commands and sub-functions added. A full implementation of the IF-THEN-ELSE logic is included, allowing nesting to 255 levels. Strings are supported and a subset of the usual string functions such as LEFT$, RIGHT$, MID$, STRING$, etc. are included. Variables are dynamically allocated. Also included are additional features such as Peek and Poke. A must for any Color Computer user running OS-9.

CoCo OS-9 $39.95

C Compiler from Windrush Micro Systems by James McCosh. Full C for FLEX, SK-DOS except bit-fields, including an Assembler. *Requires the TSC Relocating Assembler if user desires to implement his own Libraries.*

FLEX, SK-DOS, CCF - $295.00

C Compiler from Introl -- Full C except Doubles and Bit Fields, streamlined for the 6809. Reliable Compiler; FAST, efficient Code. More UNIX Compatible than most.

FLEX, SK-DOS, CCF, OS-9 (Level II ONLY), UniFLEX - $575.00

PASCAL Compiler from Lucidata -- ISO Based P-Code Compiler. Designed especially for Microcomputer Systems. Allows linkage to Assembler Code for maximum flexibility.

FLEX, SK-DOS and CCF - $190.00

OmegaSoft PASCAL from Certified Software -- Extended Pascal for systems and real-time programming.

Native 68000/68020 Compiler, $575 for base package, options available. For OS-9/68000 and PDOS host system.

6809 Cross Compiler (OS-9/68000 host) $700 for complete package.

KBASIC - from S.E. MEDIA -- A "Native Code" BASIC Compiler which is now Fully TSC XBASIC compatible. The compiler compiles to Assembly Language Source Code. A NEW, streamlined, Assembler is now included allowing the assembly of LARGE Compiled K-BASIC Programs. Conditional assembly reduces Run-time package.

FLEX, SK-DOS, CCF, OS-9 Compiler /Assembler $99.00

CRUNCH COBOL from S.E. MEDIA -- Supports large subset of ANSII Level 1 *COBOL* with many of the useful Level 2 features. Full FLEX, SK-DOS File Structures, including Random Files and the ability to process Keyed Files. Segment and link large programs at runtime, or implemented as a set of overlays. The System requires 56K and CAN be run with a single Disk System. *A very popular product.*

FLEX, SK-DOS, CCF - $99.95

FORTH from Stearns Electronics -- A CoCo FORTH Programming Language. Tailored to the CoCo! Supplied on Tape, transferable to disk. Written in FAST ML. Many CoCo functions (Graphics, Sound, etc.). Includes an Editor, Trace, etc. Provides CPU Carry Flag accessibility, Fast Task Multiplexing, Clean Interrupt Handling, etc. for the "Pro". Excellent "Learning" tool!

Color Computer ONLY - $58.95

FORTHBUILDER is a stand-alone target compiler (crosscompiler) for producing custom Forth systems and application programs. All of the 83-standard defining words and control structures are recognized by FORTHBUILDER.

FORTHBUILDER is designed to behave as much as possible like a resident Forth interpreter/compiler, so that most of the established techniques for writing Forth code can be used without change.

Like compilers for other languages, FORTHBUILDER can operate in "batch mode".

The compiler recognizes and emulates target names defined by CONSTANT or VARIABLE and is readily extended with "compile-time" definitions to emulate specific target words.

FORTHBUILDER is supplied as an executable command file configured for a specific host system and target processor. Object code produced from the accompanying model source code is royalty-free to licensed users.

FLEX, CCF, SK-DOS - $99.95

EDITORS & WORD PROCESSING

JUST from S.E. Media -- Text Formatter developed by Ron Anderson; for Dot Matrix Printers, provides many unique features. Output "Formatted" Text to the Display. Use the FPRINT.CMD supplied for producing multiple copies of the "Formatted" Text on the Printer INCLUDING IMBEDDED PRINTER COMMANDS (very useful at other times also, and worth the price of the program by itself). "User Configurable" for adapting to other Printers (comes set up for Epson MX-80 with Graftrax); up to ten (10) imbedded "Printer Control Commands". Compensates for a "Double Width" printed line. Includes the normal line width, margin, indent, paragraph, space, vertical skip lines, page length, page numbering, centering, fill, justification, etc. Use with PAT or any other editor.

* Now supplied as a two disk set:

Disk #1: JUST2.CMD object file,
JUST2.TXT PL9 source:FLEX, SK-DOS - CCF
Disk #2: JUSTSC object and source in C:
FLEX, SK-DOS, OS-9, CCF

The JTSC and regular JUST C source are two separate programs. JTSC compiles to a version that expects TSC Word Processor type commands, (.pp .sp .ce etc.) Great for your older text files. The C

Availability Legends
O = OS-9, S = SK*DOS
F = FLEX, U = UniFLEX
CCO = Color Computer OS-9
CCF = Color Computer FLEX

VISA

South East Media

5900 Cassandra Smith Rd. - Hixson, Tn. 37343

Telephone: (615) 842-4600 FAX (615) 842-7990

master card

** Shipping **
Add 2% U.S.A. (min. $2.50)
Foreign Surface Add 5%
Foreign Airmail Add 10%
Or C.O.D. Shipping Only

*OS-9 is a Trademark of Microware and Motorola-*FLEX and UniFLEX are Trademarks of Technical Systems Consultants-*SK*DOS is a Trademark of Star-K Software Systems Corp.

Telephone: (615) 842-4600

South East Media

OS-9, UniFLEX, FLEX, SK*DOS SOFTWARE

Fax: (615) 842-7990

source compiles to a standard syntax JUST.CMD object file. Using JUST syntax (,p ,u ,y etc.) With all JUST functions plus several additional printer formatting functions. Reference the JUSTSC C source. For those wanting an excellent BUDGET PRICED word processor, with features none of the others have. This is it!

Disk (1) - PL9 FLEX only- FLEX, SK-DOS & CCF - $49.95
Disk Set (2) - FLEX, SK-DOS & CCF & OS-9 (C version) - $69.95
OS-9 68K000 complete with Source - $79.95

PAT from S.E. Media - A full feature screen oriented TEXT EDITOR with all the best of "PIE™". For those who swore by and loved only PIE, this is for you! All PIE features and much more! Too many features to list. And if you don't like these, change or add your own. PL-9 source furnished. "C" source available soon. Easily configured to your CRT, with special config section.

Regular FLEX, SK-DOS $129.50
• SPECIAL INTRODUCTION OFFER • $79.95
SPECIAL PAT/JUST COMBO (with source)
FLEX, SK-DOS $99.95
OS-9 68K Version $229.00
SPECIAL PAT/JUST COMBO 68K $249.00

Note: *JUST in "C" source available for OS-9*

CEDRIC from S.E. Media - A screen oriented TEXT EDITOR with availability of 'MENU' aid. Macro definitions, configurable 'permanent definable MACROS' - all standard features and the fastest 'global' functions in the west. A simple, automatic terminal config program makes this a real 'no hassel' product. Only 6K in size, leaving the average system over 165 sectors for text buffer - appx. 14,000 plus of free memory! Extra fine for programming as well as text.

FLEX, SK-DOS $69.95

BAS-EDIT from S.E. Media - A TSC BASIC or XBASIC screen editor. Appended to BASIC or XBASIC, BAS-EDIT is transparent to normal BASIC/XBASIC operation. Allows editing while in BASIC/XBASIC. Supports the following functions: OVERLAY, INSERT and DUP LINE. Make editing BASIC/XBASIC programs SIMPLE! A GREAT time and effort saver. Programmers love it! NO more retyping entire lines, etc. Complete with over 25 different CRT terminal configuration overlays.

FLEX, CCF, SK-DOS $39.95

SCREDITOR III from Windrush Micro Systems -- Powerful Screen-Oriented Editor/Word Processor. Almost 50 different commands; over 300 pages of Documentation with Tutorial. Features Multi-Column display and editing, "decimal align" columns (AND add them up automatically), multiple keystroke macros, even/odd page headers and footers, imbedded printer control codes, all justifications, "help" support, store common command series on disk, etc. Use supplied "set-ups", or remap the keyboard to your needs. Except for proportional printing, this package will DO IT ALL!

6800 or 6809 FLEX, SK-DOS or SSB-DOS, OS-9 - $175.00

SPELLB "Computer Dictionary" from S.E. Media -- **OVER 150,000 words!**

Look up a word from within your Editor or Word Processor (with the SPELCMD Utility which operates in the FLEX, SK-DOS UCS). Or check and update the Text after entry; ADD WORDS to the Dictionary, "Flag" questionable words in the Text, "View a word in context" before changing or ignoring, etc. SPELLB first checks a "Common Word Dictionary", then the normal Dictionary, then a "Personal Word List", and finally, any "Special Word List" you may have specified. SPELLB also allows the use of Small Disk Storage systems.

FLEX, SK-DOS and CCF - $129.95

STYLO-GRAPH from Great Plains Computer Co. -- A full-screen oriented WORD PROCESSOR -- (uses the 51 x 24 Display Screens on CoCo FLEX/SK-DOS, or PBJ Wordpak). Full screen display and editing; supports the Daisy Wheel proportional printers.

NEW PRICES 6809 CCF and CCO - $99.95,
FLEX, SK-DOS or OS-9 - $179.95, UniFLEX- $299.95

STYLO-SPELL from Great Plains Computer Co. -- Fast Computer Dictionary. Complements Stylograph.

NEW PRICES 6809 CCF and CCO - $69.95,
FLEX, SK-DOS or OS-9 - $99.95, UniFLEX- $149.95

STYLO-MERGE from Great Plains Computer Co. -- Merge Mailing List to "Form" Letters, Print multiple Files, etc., through Stylo.

NEW PRICES 6809 CCF and CCO - $59.95,
FLEX, SK-DOS or OS-9 - $79.95, UniFLEX- $129.95

STYLO-PAK --- Graph + Spell + Merge Package Deal!!!

FLEX, SK-DOS or OS-9 - $329.95, UniFLEX - $549.95
OS-9 68000 $695.00

DATABASE ACCOUNTING

XDMS from Westchester Applied Business Systems

FOR 6809 FLEX or SK-DOS (5/8")

Up to 32 groups/fields per record! Up to 12 character file names! Up to 1024 byte records! User defined screen and print control! Process files! Form files! Conditional execution! Process chaining! Upward/Downward file linking! File joining! Random file virtual paging! Built in utilities! Built in text line editor! Fully session oriented! Enhanced forms! Boldface, Double width, Italics and Underline supported! Written in compact structured assembler! Integrated for FAST execution!

XDMS-IV Data Management System

XDMS-IV is a brand new approach to data management. It not only permits users to describe, enter and retrieve data, but also to process entire files producing customized reports, screen displays and file output. Processing can consist of any of a set of standard high level functions including record and field selection, sorting and aggregation, lookups in other files, special processing of record subsets, custom report formatting, totaling and subtotaling, and presentation of up to three related files as a "database" on user defined output reports.

POWERFUL COMMANDS!

XDMS-IV combines the functionality of many popular DBMS software systems with a new easy to use command set into a single integrated package. We've included many new features and commands including a set of general file utilities. The processing commands are Input-Process-Output (IPO) which allows almost instant implementation of a process design.

SESSION ORIENTED!

XDMS-IV is session oriented. Enter "XDMS" and you are in instant command of all the features. No more waiting for a command to load in from disk! Many commands are immediate, such as CREATE (file definition), UPDATE (file editor), PURGE and DELETE (utilities). Others are process conunands which are used to create a user process which is executed with a RUN command. Either may be entered into a "process" file which is executed by an EXECUTE statement. Processes may execute other processes, or themselves, either conditionally or unconditionally. Menus and screen prompts are easily coded, and entire user applications can be run without ever leaving XDMS-IV

Availability Legends
O = OS-9, S = SK*DOS
F = FLEX, U = UniFLEX
CC9 = Color Computer OS-9
CCF = Color Computer FLEX

VISA

South East Media
5900 Cassandra Smith Rd. - Hixson, Tn. 37343
Telephone: (615) 842-4600 FAX (615) 842-7990

master card

** Shipping **
Add 2% U.S.A. (min. $2.50)
Foreign Surface Add 5%
Foreign Airmail Add 10%
Or C.O.D. Shipping Only

*OS-9 is a Trademark of Microware and Motorola-*FLEX and UniFLEX are Trademarks of Technical Systems Consultants-*SK*DOS is a Trademark of Star-K Software Systems Corp.

Telephone: (615) 842-4600

South East Media

OS-9, UniFLEX, FLEX, SK*DOS SOFTWARE

Fax: (615) 842-7990

IT'S EASY TO USE!

XDMS-IV keeps data management simple! Rather than design a complex DBMS which hides the true nature of the data, we kept XDMS-IV file oriented. The user view of data relationships is presented in reports and screen output, while the actual data resides in easy to maintain files. This aspect permits customized presentation and reports without complex redefinition of the database files and structure. XDMS-IV may be used for a wide range of applications from simple record management systems (addresses, inventory ...) to integrated database systems (order entry, accounting...)

The possibilities are unlimited...

FOR 6809 FLEX or SK-DOS(5"/8" Disk) $249.95

UTILITIES

Basic09 XRef from S.E. Media -- This Basic09 Cross Reference Utility is a Basic09 Program which will produce a "pretty printed" listing with each line numbered, followed by a complete cross referenced listing of all variables, external procedures, and line numbers called. Also includes a Program List Utility which outputs a fast "pretty printed" listing with line numbers. Requires Basic09 or RunB.

OS-9 & CCO object only -- $39.95; with Source - $79.95

BTree Routines - Complete set of routines to allow simple implementation of keyed files - *for your programs* - running under Basic09. A real time saver and should be a part of every serious programmers tool-box.

OS-9 & CCO object only - $89.95

Lucidata PASCAL UTILITIES (Requires Pascal ver 3)

XREF -- produce a Cross Reference Listing of any text; oriented to Pascal Source.

INCLUDE -- Include other Files in a Source Text, including Binary - unlimited nesting.

PROFILER -- provides an Indented, Numbered, "Structogram" of a Pascal Source Text File; view the overall structure of large programs, program integrity, etc. Supplied in Pascal Source Code; requires compilation.

FLEX, SK-DOS, CCF --- EACH 5" - $40.00, 8" - $50.00

DUB from S.E. Media -- **A UniFLEX BASIC decompiler** Re-Create a Source Listing from UniFLEX Compiled basic Programs. Works with ALL Versions of 6809 UniFLEX basic.

UniFLEX - $219.95

LOW COST PROGRAM KITS from Southeast Media The following kits are available for FLEX, SK-DOS on either 5" or 8" Disk.

1. **BASIC TOOL-CHEST $29.95**
   BLISTER.CMD: pretty printer
   LINEXREF.BAS: line cross-referencer
   REMPAC.BAS, SPCPAC.BAS, COMPAC.BAS: remove superfluous code
   STRIP.BAS: superfluous line-numbers stripper
2. **FLEX, SK-DOS UTILITIES KIT $39.99**
   CATS. CMD: alphabetically-sorted directory listing
   CATD.CMD: date-sorted directory listing
   COPYSORT.CMD: file copy, alphabetically
   COPYDATE.CMD: file copy, by date-order
   FILEDATE.CMD: change file creation date
   INFO.CMD (& INFOGMX.CMD): tells disk attributes & contents
   RELINK.CMD (& RELINK82): re-orders fragmented free chain
   RESQ.CMD: undeletes (recovers) a deleted file
   SECTORS.CMD: show sector order in free chain
   XL.CMD: super text lister
3. **ASSEMBLERS/DISASSEMBLERS UTILITIES $39.95**
   LINEFEED.CMD: 'modularise' disassembler output
   MATH.CMD: decimal, hex, binary, octal conversions & tables
   SKIP.CMD: column stripper
4. **WORD - PROCESSOR SUPPORT UTILITIES $49.95**
   FULLSTOP.CMD: checks for capitalization
   BSTYCIT.BAS (.BAC): Stylo to dot-matrix printer
   NECPRINT.CMD: Stylo to dot-matrix printer filter code
5. **UTILITIES FOR INDEXING $49.95**
   MENU.BAS: selects required program from list below
   INDEX.BAC: word index
   PHRASES.BAC: phrase index
   CONTENT.BAC: table of contents
   INDXSORT.BAC: fast alphabetic sort routine
   FORMATER.BAC: produces a 2-column formatted index
   APPEND.BAC: append any number of files
   CHAR.BDV: line reader

BASIC09 TOOLS consist of 21 subroutines for Basic09. 6 were written in C Language and the remainder in assembly. All the routines are compiled down to native machine code which makes them fast and compact.

1. **CFILL** -- fills a string with characters
2. **DPEEK** -- Double peek
3. **DPOKE** -- Double poke
4. **FPOS** -- Current file position
5. **FSIZE** -- File size
6. **FTRIM** -- removes leading spaces from a string
7. **GETPR** -- returns the current process ID
8. **GETOPT** -- gets 32 byte option section
9. **GETUSR** -- gets the user ID
10. **GTIME** -- gets the time
11. **INSERT** -- insert a string into another
12. **LOWER** -- converts a string into lowercase
13. **READY** -- Checks for available input
14. **SETPRIOR** -- changes a process priority
15. **SETUSR** -- changes the user ID
16. **SETOPT** -- set 32 byte option packet
17. **STIME** -- sets the time
18. **SPACE** -- adds spaces to a string
19. **SWAP** -- swaps any two variables
20. **SYSCALL** -- system call
21. **UPPER** -- converts a string to uppercase

For OS-9 - $44.95 - Includes Source Code

SOFTOOLS

The following programs are included in object form for immediate application. PL/9 source code available for customization.

READ-ME Complete instructions for initial set-up and operation. Can even be printed out with the included text processor.

CONFIG one time system configuration.

CHANGE changes words, characters, etc. globally to any text type file.

CLEANTXT converts text files to standard FLEX, SK-DOS files.

COMMON compare two text files and reports differences.

COMPARE another check file that reports mis-matched lines.

CONCAT similar to FLEX, SK-DOS append but can also list files to screen.

DOCUMENT for PL/9 source files. Very useful in examining parameter passing aspects of procedures.

Availability Legends
O = OS-9, S = SK*DOS
F = FLEX, U = UniFLEX
CCO = Color Computer OS-9
CCF = Color Computer FLEX

VISA

South East Media

5900 Cassandra Smith Rd. - Hixson, Tn. 37343

Telephone: (615) 842-4600 FAX (615) 842-7990

mastercard

** Shipping **
Add 2% U.S.A. (min. $2.50)
Foreign Surface Add 5%
Foreign Airmail Add 10%
Or C.O.D. Shipping Only

*OS-9 is a Trademark of Microware and Motorola-*FLEX and UniFLEX are Trademarks of Technical Systems Consultants-*SK*DOS is a Trademark of Star-K Software Systems Corp.

Telephone: (615) 842-4600

South East Media

OS-9, UniFLEX, FLEX, SK*DOS SOFTWARE

Fax: (615) 842-7990

ECHO echos to either screen or file.
FIND an improved *find* command with "*pattern*" matching and wildcards. Very useful.
HEX dumps files in both hex and ASCII.
INCLUDE a file copy program that will accept "includes" of other disk files.
KWIC allows rotating each word, on each line to the beginning. Very useful in a sort program, etc.
LISTDIR a directory listing program. Not super, but better than CAT.
MEMSORT a high-speed text file sorter. Up to 10 fields may be sorted. Very fast. Very useful.
MULTICOL width of page, number of columns may be specified. A MUST!
PAGE similar to LIST but allows for a page header, page width and depth. Adjust for CRT screen or printer as set up by CONFIG. A very smart print driver. Allows printer control commands.
REMOVE a fast file deleter. Careful, no prompts issued. Zap, and its gone!
SCREEN a screen listing utility. Word wraps text to fit screen. Screen depth may be altered at run time.
SORT a super version of MEMSORT. Ascending/descending order, up to 10 keys, case over-ride, sort on nth word and sort on characters if file is small enough, sorts in RAM. If large file, sort is constrained to size of your largest disk capacity.
TPROC a small but nice text formatter. This is a complete formatter and has functions not found in other formatters.
TRANSLIT sorts a file by x keyfields. Checks for duplications. Up to 10 key files may be used.
UNROTATE used with KWIC this program reads an input file and unfolds it a line at a time. If the file has been sorted each word will be presented in sequence.
WC a word count utility. Can count words, characters or lines.

NOTE: this set of utilities consists of 6 5-1/4" disks or 2 8" disks, with source (PL9). 3 5-1/4" disks or 1 8" disk without source.
Complete set SPECIAL INTRO PRICE:
5-1/4" with source FLEX or SK-DOS - $129.95
without source - $79.95
8" with source - $79.95 - without source $49.95

FULL SCREEN FORMS DISPLAY from Computer Systems Consultants - - TSC Extended BASIC program supports any Serial Terminal with Cursor Control or Memory-Mapped Video Displays; substantially extends the capabilities of the Program Designer by providing a table-driven method of describing and using Full Screen Displays.
FLEX, SK-DOS and CCF. UniFLEX - $25.00, with Source - $50.00

SOLVE from S.E. Media - OS-9 Levels I and II only. A Symbolic Object/ Logic Verification & Examine debugger. Including inline debugging, disassemble and assemble. SOLVE IS THE MOST COMPLETE DEBUGGER we have seen for the 6809 OS-9 series! SOLVE does it all! With a rich selection of monitor, assembler, disassembler, environmental, execution and other miscellaneous commands, SOLVE is the MOST POWERFUL tool-kit item you can own! Yet, SOLVE is simple to use! With complete documentation, a snap! Everyone who has ordered this package has raved! See review - 68 Micro Journal - December 1985. No 'blind' debugging here, full screen displays, rich and complete in information presented. Since review in 68 Micro Journal, this is our fastest mover!
Levels I & II only - OS-9 **$69.95**

DISK UTILITIES

OS-9 VDisk from S.E. Media -- For Level I only. Uses the Extended Memory capability of your SWTPC or Gimix CPU card (or similar format DAT) for FAST Program Compiles, CMD execution, high speed inter-process communications (without pipe buffers), etc. - SAVE that System Memory. Virtual Disk size is variable in 4K increments up to 960K. Some Assembly Required.
Level I OS-9 object $79.95; with Source $149.95

O-F from S.E. Media -- Written in BASIC09 (with Source), includes: REFORMAT, a BASIC09 Program that reformats a chosen amount of an OS-9 disk to FLEX, SK-DOS Format so it can be used normally by FLEX, SK-DOS; and FLEX, a BASIC09 Program that does the actual read or write function to the special O-F Transfer Disk; user-friendly menu driven. Read the FLEX, SK-DOS Directory, Delete FLEX, SK-DOS Files, Copy both directions, etc. FLEX, SK-DOS users use the special disk just like any other FLEX, SK-DOS disk
OS-9 - 6809 $79.95

LSORT from S.E. Media - A SORT/MERGE package for OS-9 (Level I & II only). Sorts records with fixed lengths or variable lengths. Allows for either ascending or descending sort. Sorting can be done in either ASCII sequence or alternate collating sequence. Right, left or no justification of data fields available. LSORT includes a full set of comments and errors messages.
OS-9 $85.00

HIER from S.E. Media - *HIER is a modern hierarchal storage system for users under FLEX, SK-DOS.* It answers the needs of those who have hard disk capabilities on their systems, or many files on one disk - any size. **Using HIER a regular (any) FLEX, SK-DOS disk (8 - 5 - hard disk) can have sub directories.** By this method the problems of assigning unique names to files is less burdensome. Different files with the exact same name may be on the same disk, as long as they are in different directories. For the winchester user this becomes a must. Sub-directories are the modern day solution that all current large systems use. **Each directory looks to FLEX, SK-DOS like a regular file, except they have the extension '.DIR'.** A full set of directory handling programs are included, making the operation of HIER simple and straightforward. A special install package is included to install HIER to your particular version of FLEX, SK-DOS. Some assembly required. Install indicates each byte or reference change needed. Typically - 6 byte changes in source (furnished) and one assembly of HIER is all that is required. No programming required!
FLEX - SK-DOS $79.95

COPYMULT from S.E. Media -- Copy LARGE Disks to several smaller disks. FLEX, SK-DOS utilities allow the backup of ANY size disk to any SMALLER size diskettes (Hard Disk to floppies, 8" to 5", etc.) by simply inserting diskettes as requested by COPYMULT. No fooling with directory deletions, etc. COPYMULT.CMD understands normal "copy" syntax and keeps up with files copied by maintaining directories for both host and receiving disk system. Also includes BACKUP.CMD to download any size "random" type file; RESTORE.CMD to restructure copied "random" files for copying, or recopying back to the host system; and FREELINK.CMD as a "bonus" utility that "relinks" the free chain of floppy or hard disk, eliminating fragmentation.
Completely documented Assembly Language Source files included. ALL 4 Programs (FLEX, SK-DOS, 8" or 5") $99.50

Availability Legends
O = OS-9, $ = SK*DOS
F, = FLEX, U = UniFLEX
CC9 = Color Computer OS-9
CCF = Color Computer FLEX

VISA

South East Media

5900 Cassandra Smith Rd. - Hixson, Tn. 37343
Telephone: (615) 842-4600 FAX (615) 842-7990

mastercard

** Shipping **
Add 2% U.S.A. (min. $2.50)
Foreign Surface Add 5%
Foreign Airmail Add 10%
Or C.O.D. Shipping Only

*OS-9 is a Trademark of Microware and Motorola. *FLEX and UniFLEX are Trademarks of Technical Systems Consultants. *SK*DOS is a Trademark of Star-K Software Systems Corp.

Telephone: (615) 842-4600

South East Media

OS-9, UniFLEX, FLEX, SK*DOS SOFTWARE

Fax: (615) 842-7990

COPYCAT from Lucidata -- *Pascal NOT required.* Allows reading TSC Mini-FLEX, SK-DOS, SSB-DOS68, and Digital Research CP/M Disks while operating under SK-DOS , FLEX1.O, FLEX 2.O, or FLEX 9.O with 6800 or 6809 Systems. COPYCAT will not perform miracles, but, between the program and the manual, you stand a good chance of accomplishing a transfer. Also includes some Utilities to help out. Programs supplied in Modular Source Code (Assembly Language) to help solve unusual problems.
FLEX, SK-DOS and CCF 5"-$50.00 FLEX, SK-DOS 8"-$65.00

VIRTUAL TERMINAL from S.E. Media - Allows one terminal to do the work of several. The user may start as many as eight tasks on one terminal, under *VIRTUAL TERMINAL* and switch back and forth between tasks at will. No need to exit each one; just jump back and forth. Complete with configuration program. The best way to keep up with those background programs.
6809 OS-9 & CCO - object only - $49.95

FLEX, SK-DOS DISK UTILITIES from Computer Systems Consultants -- Eight (8) different Assembly Language (with Source Code) FLEX, SK-DOS Utilities for every FLEX, SK-DOS Users Toolbox: Copy a File with CRC Errors; Test Disk for errors; Compare two Disks; a fast Disk Backup Program; Edit Disk Sectors; Linearize Free-Chain on the Disk; print Disk Identification; and Sort and Replace the Disk Directory (in sorted order). -- PLUS -- Ten XBASIC Programs including: A BASIC Resequencer with EXTRAs over "RENUM" like check for missing label definitions, processes Disk to Disk instead of in Memory, etc. Other programs Compare, Merge, or Generate Updates between two BASIC Programs, check BASIC Sequence Numbers, compare two unsequenced files, and 5 Programs for establishing a Master Directory of several Disks, and sorting, selecting, updating, and printing paginated listings of these files. A BASIC Cross-Reference Program, written in Assembly Language, which provides an X-Ref Listing of the Variables and Reserved Words in TSC BASIC, XBASIC, and PRECOMPILER BASIC Programs.
ALL Utilities include Source (either BASIC or A.L. Source Code).
FLEX, SK-DOS and CCF - $50.00
BASIC Utilities ONLY for UniFLEX -- $30.00

MS-DOS to FLEX Transfer Utilities to OS-9 For 68XXX and CCOS-9 Systems Now READ - WRITE - DIR - DUMP - EXPLORE FLEX & MS-DOS Disk. These Utilities come with a rich set of options allowing the transfer of text type files from/to FLEX & MS-DOS disks. *CoCo systems require the D.P. Johnson SDISK utilities and OS-9 and two drives of which one must be a "host" floppy.
**CoCo Version: $69.95 68XXX Version $99.95*

MISCELLANEOUS

TABULA RASA SPREADSHEET from Computer Systems Consultants -- TABULA RASA is similar to DESKTOP/PLAN; provides use of tabular computation schemes used for analysis of business, sales, and economic conditions. Menu-driven; extensive report-generation capabilities. Requires TSC's Extended BASIC.
FLEX, SK-DOS and CCF, UniFLEX- $50.00, with Source - $100.00

DYNACALC -- Electronic Spread Sheet for the 6809 and 68000.
UniFLEX- $395.00, FLEX, SK-DOS, OS-9 and SPECIAL CCF - $250.00
OS-9 68K - $299.00

FULL SCREEN INVENTORY/MRP from Computer Systems Consultants Use the Full Screen Inventory System/Materials Requirement Planning for maintaining inventories. Keeps item field file in alphabetical order for easier inquiry. Locate and/or print records matching partial or complete item, description, vendor, or attributes; find backorder or below stock levels. Print-outs in item or vendor order. MRP capability for the maintenance and analysis of Hierarchical assemblies of items in the inventory file. Requires TSC's Extended BASIC.
FLEX, SK-DOS and CCF, UniFLEX - $50.00, with Source - $100.00

FULL SCREEN MAILING LIST from Computer Systems Consultants -- The Full Screen Mailing List System provides a means of maintaining simple mailing lists. Locate all records matching on partial or complete name, city, state, zip, or attributes for Listings or Labels, etc. Requires TSC's Extended BASIC.
FLEX, SK-DOS and CCF, UniFLEX- $50.00, with Source - $100.00

DIET-TRAC Forecaster from S.E. Media -- An XBASIC program that plans a diet in terms of either calories and percentage of carbohydrates, protein and fats (C P G%) or grams of Carbohydrate. Protein and Fat food exchanges of each of the six basic food groups (vegetable, bread, meat, skim milk, fruit and fat) for a specific individual. Sex, Age, Height, Present Weight, Frame Size, Activity Level and Basal Metabolic Rate for normal individual are taken into account. Ideal weight and sustaining calories for any weight of the above individual are calculated. Provides number of days and daily calendar after weight goal and calorie plan is determined.
FLEX, SK-DOS - $59.95, UniFLEX - $89.95

GAMES

RAPIER - 6809 Chess Program from S.E. Media -- Requires FLEX, SK-DOS and Displays on Any Type Terminal. Features: Four levels of play. Swap side. Point scoring system. Two display boards. Change skill level. Solve Checkmate problems in 1-2-3-4 moves. Make move and swap sides. Play white or black. This is one of the strongest CHESS programs running on any microcomputer, *estimated USCF Rating 1600+ (better than most 'club' players at higher levels)*
FLEX, SK-DOS and CCF - $79.95
NEW

MS-DOS/FLEX Transfer Utilities For 68XXX and CoCo* OS-9 Systems. Now Read, Write, DIR, Dump and Explore FLEX & MS-DOS Disks. Supplied with a rich set of options to explore and transfer text type files from/to FLEX and MS-DOS disks. *CoCo OS-9 requires SDISK utilities & two floppy drives.
CCO $69.95 68XXX OS-9 $99.95

MS-DOS and Macintosh Software at Discounted Prices

"Call for prices, it"ll be worth the savings."

(615) 842-4600

FAX (615)842-7990

Availability Legends
O = OS-9, S = SK*DOS
F = FLEX, U = UniFLEX
CCO = Color Computer OS-9
CCF = Color Computer FLEX

VISA

South East Media
5900 Cassandra Smith Rd. - Hixson, Tn. 37343
Telephone: (615) 842-4600 FAX (615) 842-7990

MasterCard

** Shipping **
Add 2% U.S.A. (min. $2.50)
Foreign Surface Add 5%
Foreign Airmail Add 10%
Or C.O.D. Shipping Only

*OS-9 is a Trademark of Microware and Motorola-*FLEX and UniFLEX are Trademarks of Technical Systems Consultants-*SK*DOS is a Trademark of Star-K Software Systems Corp.

SOFTWARE USER NOTES

A Tutorial Series

By : Ronald W Anderson
3540 Sturbridge Court
Ann Arbor, MI 48105

From Basic Assembler to HLL's

Printers and Text Formatting

A few weeks ago I had an idea. It cost me at least a week of my spare time, but I've learned a great deal. The idea was a simple one. I had long ago written a simple text formatter called JUST. It has been advertized here and sold alone and with PAT, but that's beside the point. The original version was set up for "monospaced" printing on an Epson or other printer that uses a standard 10 or 12 characters per inch. It would work with either, but in order to configure it for different printers, the user had to understand his printer and have an Assembler. The idea was to prepare a short file that had to be assembled and appended to the JUST file to complete the configuration for the printer. I had worked out files for the MX-80, RX-80 and more recently the LX-800. Somewhere along the way in the process of adding features, I put in the ability to switch printer modes mid-line (to switch to italics, underline, emphasized or double strike mode.)

Later I had the idea that it wouldn't be too difficult to center titles done in double wide mode. Anyway, I had decided to go back and clean the software up and in the process make it so it could read a printer configuration file that could be prepared using only an editor. That is, the configuration file would contain only printable characters. I reported last month on the difficulties I got into one evening trying to add a small feature to the reworked program.

I have a printer (Centronics 737 which also was sold as an Atari 825), that has a proportionally spaced font. Some time ago I gave a copy of JUST to Lane Lester, who then was very active in 6809 computers and FLEX. He adapted it to run on a proportional spacing printer of a different kind and sent me the source listing of his modified version. I adapted it to my Centronics 737. It uses a character width table to calculate how many dot spaces must be added between words to fill a line of type in the fully justified mode. I had decided later that I could assign a width to double wide characters and to the narrow ones as well, and I ought to be able to justify lines with any of the available type fonts. I was able to get that working, but the program had grown rather than having been designed.

I was able to get that program running with a separate printer configuration file also, and I had hopes of using it on the Epson style printers as well by assigning a constant character width and by using graphics mode to insert variable dot spaces (increments of 1/120 inch) between words in addition to the spaces, to justify lines.

I was disappointed to find that the Epson had some peculiarities. The largest of these is that (with the LX-800) when you print in Near Letter Quality mode, each line is printed twice. That is, half of the character is formed on the first pass, and the second fills in the blank spaces so to speak. That slows the printing down, but it is not intolerable for the gain of having a nice near letter quality hard copy. Well, when I tried printing a word, a space, and then switching to graphics mode to print 5 dot spaces, for example, I found that the printer would print a word, go back and overprint it with the second half of the dots, back up and "print" the dot spaces, back up and print them again, and then go on to the next word. Of course, that made it take ten minutes to print a page, but it certainly was pretty when I was done. I found that the Epson wouldn't switch from NLQ to Narrow mode directly, I had to go to draft mode first. The centronics had no such problem, and it was set up for printing dot spaces between words. It does nicely justified printing in one pass per line.

Being persistent, I wasn't satisfied. I had looked at the Epson LX-800 manual and found that I could send it commands to set the left and right margins and then to fully justify a paragraph provided there were no CRs in it. I wrote a quick BASIC program to try it out and found that it works quite nicely. It would not only justify both margins, but it has other modes that will center the text on a line or right justify it. I modified my version of JUST that was made for monospaced printing and set out to try using all the Epson features set up to do that. The first problem I found with that mode is that it is not possible to intermix wide or narrow printing. It only works in NLQ mode. That fact was documented in the manual and I realized it before I started the test.

Of course I can still underline, italicize, and emphasize the typeface, so it is not completely a waste. I set out to debug my effort and found that my line count for the page was thoroughly messed up. I discovered that when I switched justification mode to RIGHT Justify or Center, I got an extra and undocumented line feed. When I switched from either of those to full justification I got another. When I was in full justification mode for a paragraph and switched it off at the end of the paragraph, and then back on for the next, I didn't get an extra line feed. It took a couple of hours to sort out all the combinations. I had left part of my software justification built in. I would fill a line with however much would fit, and then send it to the printer. I found that when I set the margins so that a line would be 66 characters, the Epson would only let me put 65 characters on that line. Apparently it needed to put some dot spaces in, or at least thought it did. This is another undocumented feature. I adjusted my text file to one less than the spacing that I thought I needed, and I then found that I could accurately predict how much text would end up on each line so my line count could be correct.

Having gotten that far, I'm not sure it was worth the effort. I suspect there are still conditions under which I will miss the line count for a page. Of course, the Epson has a "skip perforation" mode, and if I would turn that on, it would page perfectly and I wouldn't have to try to figure out when to count a line to keep in sync with what the printer did, but then the Printer doesn't print page numbers on each page, and if I want those I have to do it my hard way without the printer automatically skipping perfs, and hoping I can keep my line count in sync with the printer.

Well, now I have three different approaches to justified printing on an Epson. The simplest is to count the characters and add extra spaces between words to make the right margin come out straight. That method eliminates the possibility of double wide or narrow printing within a line and maintaining the justification. The second method will justify with wide and narrow characters in a line but it takes ten minutes to print a page and it looks as though the printer would be worn out in a month from all the carriage motions that it goes through to print in this mode. The third lets the intelligence built into the printer do most of the work, but again has limited possibilities for mixing type sizes. None of the three are really optimum solutions. I ended up frustrated enough to drop the project and use the versions that I have for a while. I'll see which I like the best and probably use it forever. I'll probably give up the idea of reading a separate printer configuration file.

The Epson supplied justification modes are almost well done. It is too bad that Epson put so much effort into it and created something not very useful when with just a little more, it would be usable and produce very nicely justified text.

Another little problem that arose with the Epson is that double strike mode works fine in draft mode but in NLQ mode it must be done by printing a character, backspacing and printing it again. Of course when the printer does the operation it goes through gyrations, printing the second strike of a letter and the first strike of the next, and then backing up all the way to the left margin before proceeding on to the next character. It takes forever. A smart driver could be written so that it could print the line, CR without line feed, print spaces to the point of the double strike, and then print the words marked for double strike again. That would be faster but at the cost of more customization. I guess since I want the most I can get out of my printers, my only hope is to do a custom version for whatever printer I am using, and have the multiple versions for the Epsons that are discussed above.

So you won't think I am picking on Epson, the Centronics printer (which is one of the few parallel printers that I have ever seen that doesn't have a standard "Centronics Interface") has a few strange mode limitations too. I found that I can't intermix draft mode and proportional on the same line, though it will let me mix proportional. double wide (actually double wide proportional since each letter becomes twice as wide as its normal dot width) and narrow, changing freely within a line. I had wanted to be able to switch to draft mode to put a table in my text since that mode is monospaced and the table items would line up vertically. I had hoped that double strike draft mode would not look too thin compared to the proportional mode. I found that I couldn't switch. The Centronics it turns out (documented in the fine print) can't mix proportional font and draft mode font on the same line. I ended up using double wide narrow mode which is just a little wider than the ten character per inch draft mode, at about 8.5 characters per inch. That really looked quite bad!

Since I am going to end up with a different version for each printer, it appears that I wasted my time working out the means of reading the printer configuration file. I suppose I could add a switch to set up whether double strike could be done by the printer or has to be done by backspace and overprint. another to do the same thing for underline, etc. By the time I got through I'd have a nightmarishly oversized monster. Besides, I don't think I have the patience to wade through all the possibilities and flags to set up different modes of operation for everything.

I truly wish printer manufacturers could get together and do a little standardization! If there is a standard of sorts, we can thank IBM for it. Most of the dot matrix printers have a standard graphics mode with standard commands to set the dot density and drive the print head. Most printers have multiple modes. The Epson has an "Epson mode" and an "IBM mode". In the IBM mode, all of the standard IBM graphics characters may be printed. That allows screen dumps that include the boxes, smiley faces, hearts, clubs, etc. The newer printers generally have a software command to switch between the two modes, and at least a slightly different set of control codes in each of the two modes. Anything to complicate things further!

Name and Address File

I've gotten frustrated for the very last time trying to keep track of names and addresses. I have an old set of address list programs and a data file that I use once a year for Christmas cards. It runs on the old 6809 system and it is not set up for easy location of an address on demand. I decided the other day to do a quick program that would read a text file consisting of names and addresses separated by a "comment line" like this

```
John R. Jones
1234 5th Street
Anywhere, MI 49123
Phone: (123) 456-7890
*
Sam Spade
c/o Pitchfork Detective Agency
9876 5th Street
Norwalk, OH 34567
*
Norman Rockpoorly
823 Anystreet
Samstown IA,78901
*
```

I decided to let the addresses be more or less free form with regard to number of lines and whether or not a telephone number is included. I modified my FIND utility to accept a text string on the command line and to always read a file called ADDR.DAT in the root directory of drive 1. I made the search case insensitive. I then fixed the new utility which I call ADDRESS, so that it would read a whole record at a time, records being separated by asterisks. If it finds a string match anywhere in the record it prints the whole record to the screen. With the above data items I could use the utility like this. ADDRESS SPADE<CR> and the whole record in which the match is found is listed on the screen. Matches are not limited to the name line of the record. For example if I would enter ADDRESS NOR <cr> I would get the Norman Rockpoorly record and the Sam Spade listing because the NOR search string would match Norwalk as well as Norman. The feature of matching anywhere in the record is useful, and not just a nuisance. For example you could find the name of a person that you happen to remember lives in Ashtabula Mississippi, without remembering anything but the name of the city. Similarly you could find a record by the name of the company or the street.

I have about 100 names in my data file, and it finds a name within what seems like less than a second. If I search for a name like

Anderson (common in my list, of course) I'll get all of the records that match printed out to the screen.

I have visions of writing a companion utility to read the file and print each record skipping the PHONE information, to mailing labels, padding each record with blank lines to a total of 6 lines. That is what is required to print labels. It has occurred to me that with this free form approach I could insert notes after the Phone: line that would list to the screen but could be suppressed in printing labels. A small business could keep a client or customer list, or an engineering department could keep a list of sales representatives for various companies and list the categories of products sold by each. Another useful function might be to format the data from the file into a list so that a name and address book could be printed out from the data.

I'll make it a rule to add the name and address of anyone who sends me a letter to which I write a response, and keep it for a while so I won't lose it before I send promised information or whatever. I have in the past sometimes sent a letter to a reader and then a week later found a better answer to his problem only to find that I have thrown out the letter thinking that I had answered it, and along with the letter, the person's address. My solution to this in the past has been to include the address on my letter and to keep a copy of my letter in a disk file, but sometimes I get carried away cleaning up my disk and delete a file too soon. This way I will only have to clean up my name and address list. It might be a good idea to do a couple of versions that would read different data files so I could keep a business address list, a personal one, and a Micro Journal one.

I'll include the program listing for ADDRESS with this column. I might as well give you something of some little substance this time since I ranted so about the "old days" above.

```
LISTING OF FILE     ADDRESS .LST  TIME 15:45:30   DATE 01/19/89   PAGE #    1

                    * ADDRESS UTILITY FOR SK*DOS /68K
                    * BY R. W. ANDERSON 1988
                    * MODIFIED 1/89 TO IMPROVE REPORTING
                    *
                    * SYNTAX: ADDRESS NAME
                    * WHERE NAME IS THE WORD OR FRAGMENT TO BE SEARCHED FOR
                    * ALL THAT IS NEEDED IS ENOUGH TO UNIQUELY OR ALMOST UNIQUELY
                    * IDENTIFY THE RECORD THAT IS WANTED.
                    *
                    * EQUATES TO SK*DOS
                    * LIBRARY FILE IS NOT LISTED
                    *
                           OPT      LIS
                    *
000000                     ORG      $0000
000000 6002     (00004 ADRESS  BRA.S    START           GOTO START
000002 0100            VER     DC.W     $0100           VERSION NUMBER
                    *
000004 A034            START   DC       PCRLF           START ON NEW LINE
000006 49FA 0346(0034E         LEA      WFCB(PC),A4     POINT AT FILE CONTROL BLOCK
00000A A005                    DC       FOPENR          OPEN FOR READ
00000C 6650     (0005E         BNE.S    ERROR1          IF NOT ZERO ERROR OPENING FILE
                    *
                    * MAIN LOOP TO READ RECORDS AND SEARCH FOR MATCH
                    *
00000E 6152     (00062 MAIN    BSR.S    GETSTR          GET SEARCH STRING INTO BUFFER
000010 A034                    DC       PCRLF           SKIP A COUPLE OF LINES ON THE SCREEN
000012 A034                    DC       PCRLF
                    *
000014 43FA 00B8(000CE LINE    LEA      LNBUF(PC),A1    POINT AT LINE BUFFER FOR RECORD
000018 49FA 0334(0034E LINE1   LEA      WFCB(PC),A4     POINT TO SYSFCB
00001C A001                    DC       FREAD           GO READ NEXT CHAR
00001E 6632     (00052         BNE.S    ERROR
000020 12C5                    MOVE.B   D5,(A1)+        PUT IN LINE BUFFER
000022 0C05 002A               CMP.B    #'*',D5         IS IT END OF RECORD?
000026 66F0     (00018         BNE.S    LINE1           GET MORE
                    * NOW HAVE RECORD, SEE IF IT MATCHES SEARCH STRING
000028 615E     (00088         BSR.S    MATCH
00002A 4A00                    TST.B    D0
00002C 6602     (00030         BNE.S    FOUND           MATCH RETURNS NON-ZERO IN D0 IF MATC
00002E 60E4     (00014         BRA.S    LINE            ELSE GO GET MORE RECORDS
                    *
000030 49FA 009C(000CE FOUND   LEA      LNBUF(PC),A4    POINT AT START OF RECORD
000034 A034                    DC       PCRLF
000036 181C            LOOP    MOVE.B   (A4)+,D4        PREPARE TO OUTPUT CHAR
000038 0C04 002A               CMP.B    #'*',D4         END OF RECORD?
00003C 6710     (0004E         BEQ.S    ENDSTR          IF YES, DONE
00003E A033                    DC       PUTCH           ELSE OUTPUT THE CHARACTER
000040 0C04 000D               CMP.B    #$0D,D4         WAS IT CR?
000044 6606     (0004C         BNE.S    NOTCR           IF NOT SKIP
000046 183C 000A               MOVE.B   #$0A,D4         ELSE OUTPUT LF
00004A A033                    DC       PUTCH
00004C 60E8     (00036 NOTCR   BRA.S    LOOP            GET MORE CHARACTERS
00004E A034            ENDSTR  DC       PCRLF           CRLF WHEN DONE
000050 60C2     (00014         BRA.S    LINE            LOOK FOR MORE RECORDS
                    *
000052 0C2C 0008 0001  ERROR   CMP.B    #8,FCBERR(A4)


LISTING OF FILE     ADDRESS .LST  TIME 15:45:34   DATE 01/19/89   PAGE #    2

000058 6604     (0005E         BNE.S    ERROR1          NOT END OF FILE
00005A A008            ERCLS   DC       FCLOSE          IF END OF FILE WE'RE DONE
00005C A01E                    DC       WARMST          BACK TO SK*DOS
00005E A037            ERROR1  DC       PERROR          OTHER ERROR, EXIT
000060 60F8     (0005A         BRA.S    ERCLS
                    *
                    * SUBROUTINES
                    *
                    * SUBROUTINE TO GET A STRING FROM COMMAND LINE INTO
                    * A BUFFER USING PC RELATIVE ADDRESSING
                    *
000062 41FA 026A(002CE GETSTR  LEA      BUFF(PC),A0     GET POINTER TO BUFFER
000066 A02D            GET1    DC       GETNXT          GET NEXT CHAR ON COMMAND LINE
000068 0C05 000D               CMP.B    #$0D,D5         IS IT CR?
```

```
00006C 6714     [00082          BEQ.S   EXIT            IF SO, WE'RE DONE
00006E 0C05 0061        GET2    CMP.B   #'a',D5         COMPARE WITH a
000072 6D0A     (0007E          BLT.S   GET3            ASCII LESS THAN a DOESN'T NEED UPPER CASE
000074 0C05 007A                CMP.B   #'z',D5
000078 6E04     [0007E          BGT.S   GET3            ASCII GREATER THAN z DOESN'T NEED
00007A 0405 0020                SUB.B   #$20,D5         CONVERT SEARCH TO UPPER CASE
00007E 10C5             GET3    MOVE.B  D5,(A0)+        PUT IN SEARCH BUFFER
000080 60E4     I00066          BRA.S   GET1            GET MORE
000082 10BC 0000        EXIT    MOVE.B  #0,(A0)         NULL TERMINATE STRING
000086 4E75                     RTS
                        *
                        * SUBROUTINE TO SEEK A MATCH BETWEEN A SEARCH STRING
                        * AND A LINE OF TEXT.  RETURNS ZERO IN D0 IF NOT FOUND,
                        * 1 IF FOUND.
                        *
000088 41FA 0244(002CE MATCH    LEA     BUFF(PC),A0     SEARCH STRING BUFFER
00008C 43FA 0040(000CE          LEA     LNBUF(PC),A1    RECORD BUFFER
000090 2449                     MOVE.L  A1,A2           A2 KEEPS TRACK OF START OF MATCH
000092 1211             MATCH1  MOVE.B  (A1),D1         GET CHAR IN D1
                        *CONVERT RECORD CHARS TO UPPER CASE TOO
000094 0C01 0061                CMP.B   #'a',D1
000098 6D0A     [000A4          BLT.S   MATC15
00009A 0C01 007A                CMP.B   #'z',D1
00009E 6E04     [000A4          BGT.S   MATC15
0000A0 0401 0020                SUB.B   #$20,D1         MAKE UPPER CASE
0000A4 B210             MATC15  CMP.B   (A0),D1         COMPARE SEARCH WITH RECORD
0000A6 6606     (000AE          BNE.S   MATCH2          IF NOT EQUAL MOVE DOWN LINE
0000A8 5288                     ADD.L   #1,A0           IF EQUAL COMPARE NEXT CHAR OF BUFF
0000AA 5289                     ADD.L   #1,A1           NEXT CHAR OF LINE
0000AC 6010     [000BE          BRA.S   MATCH3
0000AE 528A             MATCH2  ADD.L   #1,A2           IF NOT EQUAL START AGAIN
0000B0 224A                     MOVE.L  A2,A1           START AT NEXT CHAR IN LINE
0000B2 0C11 002A                CMP.B   #'*',(A1)
0000B6 6712     (000CA          BEQ.S   NOTFND          GOT TO END OF RECORD WITHOUT A MATCH
0000B8 41FA 0214(002CE          LEA     BUFF(PC),A0
0000BC 60D4     (00092          BRA.S   MATCH1          GO AROUND AGAIN.
0000BE 0C10 0000        MATCH3  CMP.B   #0,(A0)         MATCH IF WE GOT TO END OF MATCH STRING
0000C2 66CE     (00092          BNE.S   MATCH1          ELSE CONTINUE TRYING FOR MATCH
0000C4 103C 0001                MOVE.B  #1,D0           SIGNAL MATCH
0000C8 4E75                     RTS
0000CA 4200             NOTFND  CLR.B   D0              SIGNAL NO MATCH
0000CC 4E75                     RTS
                        *
0000CE                  LNBUF   DS.B    512
```

```
LISTING OF FILE     ADDRESS .LST  TIME 15:45:40   DATE 01/19/89   PAGE #    3

0002CE                  BUFF    DS.B    128
00034E 0000 0001        WFCB    DC.B    $00,$00,$00,$01  THREE ZEROS AND DRIVE NUMBER
000352 4144 4452 0000           DC.B    "ADDR",$00,$00,$00,$00 FILENAME AND ZEROS TO MAKE 8 BYTES
00035A 4441 5400                DC.B    "DAT",$00       EXTENSION AND ZERO BYTE TO COME OUT EVEN BYTES
00035E                          DS.W    300             REST OF FCB
                                END     ADDRESS
   0 ERRORS DETECTED
```

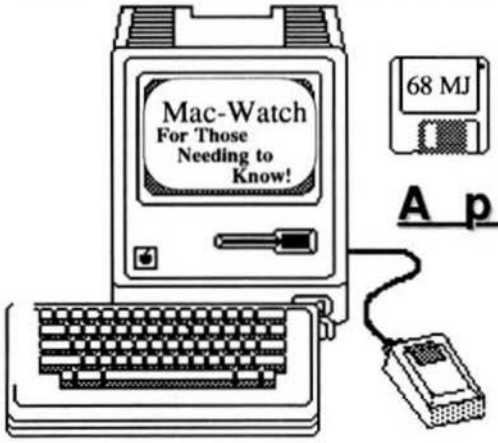
+++

```
* ADDRESS UTILITY FOR SK*DOS /68K
* BY R. W. ANDERSON 1988
* MODIFIED 1/89 TO IMPROVE REPORTING
*
* SYNTAX: ADDRESS NAME
* WHERE NAME IS THE WORD OR FRAGMENT TO BE SEARCHED FOR
* ALL THAT IS NEEDED IS ENOUGH TO UNIQUELY OR ALMOST
UNIQUELY
* IDENTIFY THE RECORD THAT IS WANTED.
*
* EQUATES TO SK*DOS
* LIBRARY FILE IS NOT LISTED
*
 LIB 0.SKEQUATE.TXT
*
 ORG $0000
ADRESS BRA.S START GOTO START
VER DC.W $0100 VERSION NUMBER
*
START DC PCRLF START ON NEW LINE
 LEA WFCB(PC),A4  POINT AT FILE CONTROL BLOCK
 DC FOPENR OPEN FOR READ
 BNE.S ERROR1 IF NOT ZERO ERROR OPENING FILE
*
* MAIN LOOP TO READ RECORDS AND SEARCH FOR MATCH
*
MAIN BSR.S GETSTR GET SEARCH STRING INTO BUFFER
 DC PCRLF  SKIP A COUPLE OF LINES ON THE SCREEN
 DC PCRLF
```

```
*
LINE LEA LNBUF(PC),A1 POINT AT LINE BUFFER FOR RECORD
LINE1 LEA WFCB(PC),A4 POINT TO SYSFCB
 DC FREAD GO READ NEXT CHAR
 BNE.S ERROR
 MOVE.B D5,(A1)+ PUT IN LINE BUFFER
 CMP.B #'*',D5 IS IT END OF RECORD?
 BNE.S LINE1 GET MORE
* NOW HAVE RECORD, SEE IF IT MATCHES SEARCH STRING
 BSR.S MATCH
 TST.B D0
 BNE.S FOUND MATCH RETURNS NON-ZERO IN D0 IF MATCH
 BRA.S LINE ELSE GO GET MORE RECORDS
*
FOUND LEA LNBUF(PC),A4 POINT AT START OF RECORD
 DC PCRLF
LOOP MOVE.B (A4)+,D4 PREPARE TO OUTPUT CHAR
 CMP.B #'*',D4 END OF RECORD?
 BEQ.S ENDSTR  IF YES, DONE
 DC PUTCH  ELSE OUTPUT THE CHARACTER
 CMP.B #$0D,D4 WAS IT CR?
 BNE.S NOTCR IF NOT SKIP
 MOVE.B #$0A,D4 ELSE OUTPUT LF
 DC PUTCH
NOTCR BRA.S LOOP GET MORE CHARACTERS
ENDSTR DC PCRLF CRLF WHEN DONE
 BRA.S LINE LOOK FOR MORE RECORDS
*
ERROR CMP.B #8,FCBERR(A4)
 BNE.S ERROR1 NOT END OF FILE
ERCLS DC FCLOSE IF END OF FILE WE'RE DONE
 DC WARMST  BACK TO SK*DOS
ERROR1 DC PERROR OTHER ERROR, EXIT
 BRA.S ERCLS
*
* SUBROUTINES
*
* SUBROUTINE TO GET A STRING FROM COMMAND LINE INTO
* A BUFFER USING PC RELATIVE ADDRESSING
*
GETSTR LEA BUFF(PC),A0 GET POINTER TO BUFFER
GET1 DC GETNXT GET NEXT CHAR ON COMMAND LINE
 CMP.B #$0D,D5 IS IT CR?
 BEQ.S EXIT IF SO, WE'RE DONE
GET2 CMP.B #'a',D5 COMPARE WITH a
 BLT.S GET3 ASCII LESS THAN a DOESN'T NEED UPPER CASE
 CMP.B #'z',D5
 BGT.S GET3 ASCII GREATER THAN z DOESN'T NEED
 SUB.B #$20,D5  CONVERT SEARCH TO UPPER CASE
GET3 MOVE.B D5,(A0)+ PUT IN SEARCH BUFFER
 BRA.S GET1 GET MORE
EXIT MOVE.B #0,(A0) NULL TERMINATE STRING
 RTS
*
* SUBROUTINE TO SEEK A MATCH BETWEEN A SEARCH STRING
* AND A LINE OF TEXT.  RETURNS ZERO IN D0 IF NOT FOUND,
* 1 IF FOUND.
*
MATCH LEA BUFF(PC),A0 SEARCH STRING BUFFER
 LEA LNBUF(PC),A1 RECORD BUFFER
 MOVE.L A1,A2 A2 KEEPS TRACK OF START OF MATCH
MATCH1 MOVE.B (A1),D1 GET CHAR IN D1
*CONVERT RECORD CHARS TO UPPER CASE TOO
 CMP.B #'a',D1
 BLT.S MATC15
 CMP.B #'z',D1
 BGT.S MATC15
 SUB.B #$20,D1 MAKE UPPER CASE
MATC15 CMP.B (A0),D1 COMPARE SEARCH WITH RECORD
 BNE.S MATCH2  IF NOT EQUAL MOVE DOWN LINE
 ADD.L #1,A0 IF EQUAL COMPARE NEXT CHAR OF BUFF
 ADD.L #1,A1 NEXT CHAR OF LINE
 BRA.S MATCH3
MATCH2 ADD.L #1,A2 IF NOT EQUAL START AGAIN
 MOVE.L A2,A1 START AT NEXT CHAR IN LINE
 CMP.B #'*',(A1)
 BEQ.S NOTFND GOT TO END OF RECORD WITHOUT A MATCH
 LEA BUFF(PC),A0
 BRA.S MATCH1 GO AROUND AGAIN.
MATCH3 CMP.B #0,(A0) MATCH IF WE GOT TO END OF MATCH
STRING
 BNE.S MATCH1 ELSE CONTINUE TRYING FOR MATCH
 MOVE.B #1,D0 SIGNAL MATCH
 RTS
NOTFND CLR.B D0 SIGNAL NO MATCH
 RTS
*
LNBUF DS.B 512
BUFF DS.B 128
WFCB DC.B $00,$00,$00,$01  THREE ZEROS AND DRIVE NUMBER
 DC.B "ADDR",$00,$00,$00,$00  FILENAME AND ZEROS TO MAKE
8 BYTES
 DC.B "DAT",$00  EXTENSION AND ZERO BYTE TO COME OUT EVEN
BYTES
 DS.W 300  REST OF FCB
 END ADRESS

+++
```

FOR THOSE WHO NEED TO KNOW **68 MICRO JOURNAL™**



The Macintosh™ Section

Reserved as

A place for your thoughts

And ours.....

Mac-Watch

By James E. Law

This month we will look at four items that will be of interest to those who use the Macintosh for desktop publishing. These include Drawing Table, DTP Advisor, Metro ImageBase clip art and Grappler C/Mac/GS.

A Review of

DRAWING TABLE

Graphics software for the Macintosh

Last month I reviewed an inexpensive page layout program and concluded that in that case, at least, you get (only) what you pay for. That's not at all to say that there aren't any real deals around. There are; you just have to look hard for them. As a matter of fact, I think I have found a real winner in a $79 (street price) draw program called Drawing Table from Broderbund Software.

When you first open Drawing Table, you won't be very impressed. You will see a row of menus, a drawing area, and a rather average drawing tool palette. Your respect for this little program will grow, however, as you discover all the extras it contains. In this review, I will assume you are familiar with the features normally provided by Macintosh graphics programs. I will primarily address the features that are not offered by competitors in the same price range.

Handling Objects

Drawing Table is a draw-type (object based) graphic program. You may import bit mapped images, but they can not be modified since no paint tools are provided. Drawing Table's tools can be chosen from a moveable and hideable tool palette. Double clicking any object brings up a specification block where you set line and fill patterns (144 choices!), foreground and background color, and line width (up to 1/2 inch). Double clicking a text block brings up a specification block where the text characteristics including font color may be set.

The tool palette contains a rotation tool that can be used to rotate any object, including text. By selecting "Show Size" from the Option menu, the size and degree of rotation for any object can be displayed. This feature is very useful in free rotation to a specific degree. The "Set" menu also allows rotation in 90 degree increments (e.g., rotate left, flip up-down).

Other tools under the "Set" menu for handling objects include "Send to Back or Front), Lock (or Unlock), Group (and Ungroup), Scale, Align, and Distribute. The "Align" option provides for 16 different ways of aligning selected items.

See It My Way

Drawing Table windows may be viewed at up to 8 times normal or as little as 1/8 of normal size. Zooming in or out may be accomplished through menu selections or by keyboard combinations. A number of different Drawing Table documents can be opened at once and automatically arranged in any one of 6 options so that any one of the document can easily be seen, selected, and made active.

Handling Text

Drawing Table allows text to be entered and edited. All text in a given text block must be the same font family and size. A nice-to-have feature is the "Case" option which allows you to easily change the case of selected text (e.g., all caps, all lower case, first letter upper case). The most outstanding text feature is the ability to bind a line of text to any path. This feature is not available in comparable priced programs that I am familiar with. To use this feature you draw a polygon (e.g., arc, circle, box), enter the desired text, select both the text and the polygon, then choose "Bind Text". Drawing Table then gives you several options for how the text will be placed (e.g., centered or justified). The text then appears along the selected path. The polygon can be made to be visible or not visible. The text cannot be edited while bound on a path but you can easily unbind, edit, then re bind

the text. This is a handy capability for creating logos, flyers, or other eye-catching publications. In a few of my trials, the letters of the text did not look evenly positioned when printed on a PLP laser printer.

User Interface

The thing that impressed me most about Drawing Table, was its well thought out user interface. For example, as your cursor approaches any object, the appearance of that object changes to let you know that it is selectable. Select any object then enter COMMAND plus the "=" key and select a different object. The first object is then replaced by a copy of the second object. In a similar manner, you can make any selected object assume the characteristic of another object (e.g., line weight, fill pattern, font family, font size). By changing the keys used in conjunction with the mouse, you can specify exactly which characteristic will be copied. While this feature will take some practice to memorize the various commands, it could be a real time saver. The feature I liked the best

An example of *Drawing Table's* ability to bind text to a path

was the "now you see it, now you don't" tool palette. The tool palette can be hidden to provide a larger, unobstructed working area. When you need it, simply press OPTION-COMMAND and click the mouse then presto, the tools palette appears right under your pointer! When you make your selection, it disappears again. The final example of an unusually well done user interface is the "temporary selection arrow". While using any tool, you may temporarily turn the cursor into the selection arrow by pressing the OPTION key. Now that's a neat feature! How many times have you been in a text entry mode and discovered the text block is misplaced or is too small. In most programs you have to go select the selection arrow tool, modify the text block, then go back to the tool palette for the text tool, get a text insertion point and continue. With Drawing Table, you just have to press and hold the OPTION key to have the cursor turn into a selection arrow. After you make your changes to the text block you can release the OPTION key and the cursor will return to its original state. All I've got to say is, this makes so much sense, why doesn't all software provide it? A minor complaint with the interface is that the selection area around objects is unusually large making it hard to select object that are close to one another.

Conclusion

I haven't come close to fully describing Drawing Table but hope I have given you a flavor for its abilities. This program have one of the smoothest user interfaces than I have worked with. Its ability to bind text along an arbitrary path is entirely unexpected for an entry-level graphics program. If you are in the market for an object based graphics program, I don't think you will be disappointed with Drawing Table.

Another relatively new product from Broderbund is DTP Advisor, a HyperCard-based desktop publishing tutorial and project management system.

The popularizing of graphics-based computers like the Macintosh gave raw desktop publishing capability to the masses.

A review of
DESKTOP PUBLISHING (DTP) ADVISOR

PageMaker, a mouse, and a laser printer, however, do not ensure quality output. To the discomfort of DTP professionals, the quantity of poorly designed and poorly produced publishing projects has proliferated. Obviously then there is potentially a big market for DTP aids and training tools.

DTP Advisor consists of several hypercard stacks. It allows non sequential study of a variety of DTP topics. The student can proceed both at his or her own pace and direction in exploring this material. Sound is not used. There is very limited use of animation. The screen images are attractive and easy to read.

The six subjects covered by the educational part of DTP Advisor include:

1. Planning - Defining your project and understanding your target audience.
2. Project Management - Ensuring that your project is well organized, on time, and on budget.
3. Design - Producing creative concepts and layouts.
4. Typography - Typesetting for visual interest and readability.
5. Art Production - Refining and assembling material for the printer.
6. Printing - Hiring a printer and understanding the printing process.

This material is introductory in nature and its targeted audience is the new comer to DTP. For example, only four Macintosh-sized windows are devoted to typography. This is not to say the material is not useful; it is. Just don't expect to become fully DTP literate by interacting with this product.

DTP Advisor provides a variety of forms for developing and managing individual DTP projects. For example, layout forms are provided which can be used with HyperCard's graphics tools to do

rough layouts. Other forms cover project definition, schedule, and estimation; typography and printing specifications; etc. A number of the forms have formulas built in (like a spreadsheet) to calculate total costs based on parameters you enter. You probably will not want to use the forms as they are (either in HyperCard or as hard copies) but they may be good sources of ideas for the design of your own forms.

DTP Advisor also contains a data base that you may use to keep track of suppliers. An entry form is provided with a variety of sort and print options.

Conclusion

If you are a newcomer to DTP and are a HyperCard fan, you may benefit from DTP Advisor and may enjoy exploring it. It does contain good information. Personally, I am not convinced that any HyperCard training I have seen so far does anything that a good book could not do better.

A REVIEW OF METRO IMAGEBASE CLIP ART

Professional electronic art from Metro ImageBase, Inc.

Clip art has come a long way since those early bit mapped images that we pasted into MacWrite documents. We were proud of the resulting works of art in spite of the jagged edges that went with 72 DPI images. Well, our expectations are higher now and variety of companies like Metro ImageBase, Inc., are doing a good job of satisfying current needs with relevant, useful, and high resolution (i.e., 300 DPI) images. Color clip art is also beginning to appear.

Metro ImageBase supplies 14 packages of 300 DPI clip art for the MacIntosh and MS-DOS environments. Each package is $145. The Mac versions are in TIFF format. Each package contains 100 images in compressed format on six 800k disks. The Unstuff it DA is provided to decompress the images. The typical image is about a quarter of a page in size, occupies 70k compressed and 250k decompressed.

I examined two packages entitled Business Graphics and Report Maker and was impressed with the quality of the art. There was a little too much repetition (five or six images of people talking on phones) but with 100 images per package, you are bound to find something to meet your need. Other reviewers have concluded that much of this art is dated but I didn't find that to be the case.

TIFF images can be modified in programs like Image Studio, Digital Darkroom, and Canvas. There are still a number of word processors, however, as well as desktop presentation programs which cannot import TIFF but I did not find this to be much of a problem. I simply opened the TIFF images in Canvas and saved them in PICT format. I could then use them with almost every program that imports graphics and they still printed sharp and crisp at 300 DPI on my non postscript laser printer. Of course, either TIFF or PICT images can be re sized (within limits) without serious loss of clarity.

Some of the other topics offered by Metro ImageBase include:

- Newsletter Maker
- The Four Seasons
- Exercise and Fitness
- Weekend Sports
- Computers and Technology
- Art Deco
- Food
- People
- Travel

Metro ImageBase clip art can be used with any Macintosh. It must be used with software that accepts TIFF images or software which can convert it to another usable format (e.g., PICT). Each package is provided in a plastic case with a users' manual and an image index.

Metro Image Base is providing a real service to the publishing professional in providing this high quality and diverse collection of clip art. If you are ready to step up from the jagged bit-mapped images of the past to take advantage of your laser printer's resolution, I recommend that you strongly consider this product.

Making the 'right move' with a full size image from the Metro ImageBase clip art collection

A Review of

Grappler C/Mac/GS

A 9 Pin Parallel Printer Interface for Macintosh

In the world of computers and printers, information can be sent or received in two forms: serial or parallel. Information that is sent serially is like a railroad train: each message is a single car that arrives separately as it comes down the track. On the other hand, imagine an eight lane super highway with eight cars traveling side by side, and all arriving simultaneously. This is how parallel information travels.

Normally, the Macintosh, Apple IIe and IIGS computers send out serial information, expecting a serial printer. The Grappler C/MAC/GS is a unique printer cable that actually converts information from serial to parallel. This conversation allows you to connect your Apple to a parallel printer instead of an ImageWriter, thus increasing the number of printer choices available to you.

I had a chance to buy a used Epson MX-80 at a really low price ($50). The only problem was that my Mac only wanted to have an ImageWriter hanging off its back. Grappler was the common denominator in my complex equation of interfacing the Epson and my Mac Plus.

The 9 Pin Parallel Printer Interface for the Apple Macintosh

This is the most transparent, easy to use interface to connect your Macintosh with your parallel printer. Why? Because it simply takes the Apple ImageWriter code form the Mac and converts it to the code your parallel printer understands. The Mac thinks it's talking to an ImageWriter and the Grappler handles the rest. The Grappler C/MAC/GS gives you Mac-like output on dozens of popular 9 pin Epson or IBM ProPrinter compatible printers including: Brother, Citizen, C.Itoh, DataProducts, Facit, Fujitsu, Centronics, Okidata, Olympia, Panasonic, Seikosha, Star, and others. The Grappler C/MAC/GS also supports color printing on color models.

Intelligent Printer Cable

The Grappler is actually an intelligent cable that links your Mac to the printer of your choice. One end of the cable is a standard mini-DIN 8 plug which fits into either your Mac's modem or printer port. The other end, which plugs into your printer's Centronics parallel port, houses the Grappler innovative electronics. You just set the Grappler's DIP switch for your printer, plug in the Grappler, and you're ready to print. The Grappler takes care of translating the ImageWriter code so your printer can understand it.

WHAT'S IN THE PACKAGE:

In addition to the manual (which is very good), you would receive the following with your Grappler C/MAC/GS:

1. One Grappler C/MAC/GS unit with Dip Switch.
2. One Apple IIc converter cable.
3. One 5-1/4" disk (Apple IIc & IIGS utilities)
4. One power supply.

The Intelligent Connection Between Macs and 24 Pin Dot Matrix or Laser Printers Grappler LQ works with many popular 24 pin printers such as NEC, Panasonic and Epson LQ, by utilizing Apple's powerful ImageWriter LQ driver. It also allows you to use your Mac with HP compatible parallel laser printers. As long as your software works with the ImageWriter LQ, it will now work with a wide variety of high quality printers, thanks to the Grappler LQ. The Grappler LQ is fully Macintosh compatible, printing both graphics and text, in portrait (vertical) or landscape (sideways).

Serial Mac Interface for HP DeskJet and Laser Printers

The Grappler LS is designed to give you freedom of choice in printer connectivity for your Macintosh. The Grappler LS connects your Mac to HP-compatible serial laser printers or the innovative HP DeskJet 300 DPI inkjet printers-retail under $1000!

Transparent To Software

Since the Grappler C/MAC/GS uses the ImageWriter II driver, it is compatible with all Macintosh software supported by Apple printers. It is also completely transparent to the user. When it's time to print you simply select the ImageWriter icon in chooser, and print. No matter which printer you use, the output you achieve is similar to that produced by Apple's ImageWriter II.

The Grappler is available from:
Orange Micro Inc.
1400 N. Lakeview Ave.
Anaheim, CA 92807

to order call:
1-800 223-8029 or for the nearest dealer near you.

FOR THOSE WHO NEED TO KNOW **68 MICRO JOURNAL™**

Bit-Bucket

By: All of us

"Contribute Nothing - Expect Nothing", DMW '86

ULTRASCIENCE SAYS THANKS

Thanks to the 68Micro Journal for being an excellent OS-9 communication port. Thanks to all of you OS-9 folks who gave our PC68K1 coprocessor such a rousing reception when it was announced last month. We knew there was a great deal of interest in placing OS-9 into the PC environment, but we were pleasantly surprised nevertheless. In appreciation for your support, we will make a maximum effort to see that the PC68K1 product remains part of the cutting edge for OS-9 development and mass marketing. A full featured version of OS-9 2.3 is now available for the PC68K1. We have also made a number of enhancements to the PC68K1 specific software. Disk access is now even faster, and high-performance 5 1/4" SCSI disk and tape devices with capacities of as much as 2 gigabytes are supported. The code that dynamically allocates time between DOS and OS-9 has been further optimized, so that persons using DOS concurrently with OS-9 will notice a significant improvement in throughput. Drivers for the twelve OS-9 serial ports have been improved to better handle process control inputs.

Unfortunately, the PC68K1 has not made everyone happy! So, for those of you who do not now own, nor probably ever will own a PC, we have a present -- OS-9 for the MAC. We hope that you will be as pleased as the PC/XT/AT users. Call or write for your specifications.

ULTRASCIENCE Div., Gibbs Laboratories, Inc.
1824 Wilmette Avenue, Wilmette, IL 60091
Telephone 312/256-0080
Fax 312/256-0097

FACETS

The variety of specific applications which can be built around the OS-9 operating system is limitless. The dense kernel and interlinking core modules strike an optimal balance between flexibility and remarkably efficient real-time performance. OS-9 also provides exceptionally powerful inter-process message passing capability, hardware independence, support for essentially all the major high-level languages, ..., and it gets better all the time. What more could you wish for?

Ultrascience provides powerful S/R Facet™ software to enhance your interface with OS-9, to facilitate rapid application design, and to extend OS-9 hardware independence for peripheral devices. The list of S/R Facet™ software is open-ended, and suggestions are always welcome.

Brief Descriptions

S/R TICTOC - is an acronym for "Terminal Input Conversion / Terminal Output Conversion." The TICTOC interface is simpler to use, yet more powerful than termcaps. TICTOC neutralizes differences between terminals, even major differences, so extensively that enhanced screens will display properly without programming changes. Using TICTOC, the same program will display correctly and brilliantly on a Wyse 50 (embedded mode), a Link (non-embedded mode), and a VT220 (character mode). TICTOC also converts input from terminals, permitting keyboards to be adapted dynamically to the requirements of an application. The unified, well ordered, standard TICTOC commands provide cursor control, visual enhancements, graphics, function and edit key translations, auxiliary port and printer controls, ... for an evergrowing library of "terminal handlers". New handlers are being created all the time, and custom handlers are easily created using the powerful TICTOC MAKER.

S/R SHELL - The Bourne Shell is largely responsible for the current success of UNIX. It is a well established, command processing, program language; complete with wildcarding, variables, pipelines, redirects, tests, structured conditionals, operators, backticks, ... Ultrascience adapted the Bourne Shell for OS-9. Powerful and easily learned, SHELL should should be a part of every OS-9 system.

S/R CRON - permits you to set up a list of functions which are to be performed automatically by the system at the time(s) you specify. You may set functions to execute once, or periodically, at a particular time-of-day, day-of-week, or month-of-year. There is no limit to the number of operations you can queue up, and they can be established for as long as a year into the future.

S/R XDIR - provides an interactive graphic display of your file directories. It displays multiple files and directories; you can walk along any directory path and get an instant view of the files in the directory. XDIR will optionally display file attributes such as size, owner, date, s/rw flags, etc. Searches for files based on a wild-card, "regular" expression, can be made recursively, and/or within selected directories. Files and directories can be marked and then used as input to any OS/9 shell command. A VTREE feature allows you to see a grapic display of your directory structure, and a special mode of XDIR makes it behave like the UNIX find, so that filename matches can be used as standard input in a pipeline command.

S/R MENU - is the quickest way to establish a friendly interface between yourself and the OS/9 system. Menu selections are defined by title, selection key, and the OS/9 system command to be executed; MENU does the rest. It balances your menu display by selection count and title length, draws pretty boxes, inserts menu headers and date/time - you select the style. A single key-stroke is all that is necessary to select an option from a completed menu. Since MENU can execute any OS/9 command, MENU can invoke another MENU, thereby permitting unlimited nesting of menus.

S/R CPIO - Moves data in and out of standard CPIO format for inter-system transfer and tape or floppy storage.

S/R TAR - Moves data in and out of standard TAR format for inter-system transfer and tape or floppy storage.

HARDWARE DIAGNOSTICS

Hundreds of gates, thousands of gates, millions of gates..... It is hard enough to find a broken one; but how does one find the intermittent one, the one which is sensitive to temperature, voltage, or simply has a mind of its own? Board swapping can be a powerful troubleshooting tool if you have sufficient redundancy in your system or systems, and sophisticated logic probes and bus analyzers can perform wonders.

However, a really good diagnosis always makes any repair easier - often trivial enough to complete in the field without test equipment. Ultrascience thought it would be nice to offer a set of do-it-yourself software tools for diagnosing hardware failures; something really easy to use, something that would make it possible to use the sophistication of an OS-9 computer to diagnose itself.

Brief Descriptions

S/R cputest - exercises the 680XX master CPU chip with an extensive battery of Motorola CPU tests (e.g.; addressing modes; arithmetic; data movement; branching instructions; exception processing; and memory management, if applicable; etc.). Pass-fail is reported.

S/R fpctest - challenges the floating point coprocessor with an array of function tests (e.g., move, fsave, frestor; status register reads and writes, exception handling for overflow, etc.), designed to detect malfunction. Pass-fail is reported.

S/R dramtest - tests DRAM with an intense series of challenges, designed to test for complex gate interaction and refresh failures, as well as simple "stuck" bits. The addresses of any errors are reported.

S/R siotest - tests any two serial I/O ports on a system against one another to confirm flow control and data integrity. All 256 ASCII characters are transferred in both directions and errors are reported. Both XON/XOFF and hardware flow control, DSR/DTR or CTS/CTR, are also tested. Errors are reported.

S/R clktest - compares the system clock to the date and time circuit. The tick rate error of the system clock with respect to the date and time circuit is reported. Run in background clktest will dynamically change the tick rate of the system clock so that the system time will track the time of the date and time circuit. This method of keeping the system time correct is superior to resetting the system clock in a step fashion. In most systems (depending upon the system clock interrupt rate) clktest will maintain the system clock within 5 seconds of the date and time circuit.

S/R tickfix - sets the system time-keeping software's tick rate.

S/R tapetest - records and compares all 256 ASCII characters in a user specified number of tape blocks. Errors are reported.

S/R disktest - performs a high-speed read of all the blocks on a disk. Any replaced blocks and newly detected, defective blocks are reported.

COMPUSENSE Ltd.

Computer Systems Consultants
68a Willoughby Lane,
London N17 0SP

Telephone 01-885 3300
Telex 8813271 COSCOM G
Fax 01-801 2640

68070 and I²C Evaluation Card plugs into IBM-PC

Compusense has launched an evaluation system for the Philips SCC68070 16/32 bit CMOS highly integrated 68000 compatible microprocessor.

The PI-68070 includes a software monitor with debugging facilities and support for the integral RS-232 and the Philips I²C bus.

The system is on a standard PC expansion card with the 68070 cpu, ROM, RAM, I²C EEPROM, the PC bus interface, a bus expansion interface and connectors for the RS-232 and the I²C bus. The bus expansion and I²C connectors allow custom hardware designs to be interfaced to the 68070 and tested.

Standard 68000 software development tools are used. A PC based cross assembler is available and compilers for languages such as C and Modula 2 can also be used. Programs are loaded from the PC via the PC interface or the RS-232.

For more details contact: Stan Opyrchal

68070 SUPPORT FOR THE PC

COMPUSENSE Ltd of London, England announce the PI-68070, the first PC processor card to use the 68070.

The 68070 is a highly integrated 68000 compatible microprocessor, produced by Philips, and features on-chip peripherals including:

- ⇨ Memory Management Unit (MMU),
- ⇨ Universal Asynchronous Receiver/Transmitter (UART),
- ⇨ Timer, Direct Memory Access (DMA) channels and
- ⇨ I²C Bus.

The 68070 is at the heart on the interactive compact disk system (CDI) being developed by Philips and Sony.

The PI-68070 supports the 68070 for general 68000 development work as well as the special features of the 68070 and the I²C bus. The I²C bus is a two wire serial bus connects a large range of inexpensive peripheral ICs produced by Philips.

For more details contact: Compusense Ltd,
68A Willoughby Lane, London N17 0SP
phone: 01-885 3300 Fax: 01-801 2640

MOTOROLA INC.

Microprocessor Products Group
6501 William Cannon Drive West
Austin, Texas 78735-8598
For further information contact:

CONTACTS
Shernaz Daver
Cunningham Communication, Inc.
(408) 982-0400

Dean Mosley
Motorola Inc.
(512) 891-2839

MOTOROLA ANNOUNCES 50 MHz 68030

Fastest Clock Speed in Industry
Double the Performance of Competing Architectures

AUSTIN, Texas, April 5, 1989 — Motorola's Microprocessor Products Group today announced the availability of a 50 MHz version of its 68030 (030) microprocessor. The new microprocessor offers the fastest clock speed in the industry, surpassing all CISC and RISC products on the market. The chip delivers 12 MIPS (million instructions per second) of performance, double that of all conventional processors available today.

The 50 MHz 030 is manufactured in 1 micron HCMOS (High-Performance Complementary Metal Oxide Silicon) technology, making it the first conventional processor to be produced below 1.2 microns. Like its predecessor, the 33 MHz 030, the new part is software compatible with all microprocessors in Motorola's 68000 family. This compatibility allows the new part to access a $3 billion 32-bit software base, started in 1979 with the introduction of the first 68000 microprocessor. The 50 Mhz version of the 030 also provides a high-performance path for a $100 billion hardware base that includes systems from Apollo Computer, Apple Computer, Hewlett-Packard, NEC, NeXT, Sony Microsystems and Sun Microsystems.

"Our 50 Mhz 030 clearly puts us a step ahead of our competitors," said Jack W. Browne, director of marketing of Motorola's Microprocessor Products Group (Austin, Texas). "The speed of this new part allows our customers to offer ultrahigh-performance systems while maintaining a huge software base."

The 50 MHz 030 enters general sampling in May. Volume production will begin in the third quarter of this year. The part is priced at $650 for sample quantities.

Last week, Motorola disclosed features of its 68040 (040) microprocessor, the newest member of the 68000 family. In addition to the 040, the 68000 family consists of the 68000, 68010, 68020 and 68030. All microprocessors in the family are software compatible, allowing for a $3 billion software base to migrate from one microprocessor to the next. The 68000 family is credited for creating the workstation and graphics markets. According to InfoCorp, a market research firm based in Santa Clara, Calif., the 68000 architecture powers more than 63 percent of all computer systems priced from $12,000 to $300,000. The 68000 microprocessors are also used in embedded control applications, personal computers and consumer systems.

Motorola's $2.74 billion Semiconductor Products Sector (Phoenix, Ariz.), which includes the Microprocessor Products Group (Austin, Texas), is a part of Motorola Inc. The company is the largest and broadest supplier of semiconductors in North America, with a balanced portfolio of more than 50,000 devices.

Windsor Systems

2407 Lime Kiln Lane, Louisville, KY 40222 U.S.A.
Telephone: 502-425-9560 Fax: 502-426-3944

The Rowley Modula-2 Compiler

Cross PC to 68030
Native Transputer
Native QL-[illegible]
Native Sun/Apollo

Windsor Systems announces the Rowley™ Modula-2 compiler. It features fast two-pass compilation and highly optimised code generation. All 68000 series microprocessors and coprocessors are supported. The Transputer Helios™ version supports mixed code with other languages. All versions give full operating system access.

Rowley™ conforms to Niklaus Wirth's Edition 3 of the language, while appreciating the pending BSI and ISO standardisation recommendations. It features LONG and SHORT types, and can address up to 2 Gigabytes of memory. The standard and run-time libraries are supplied in both object and source code form.

A set of source code tools is provided. A run time debugger and ROM tools are available. Prices are as follows: Compiler £1795.00, Debugger £720.00, ROM tools £720.00. Site licences are available.

Contact Windsor Systems for further information.
April 1989

FORCE COMPUTERS INC
3165 Winchester Blvd.
Campbell, CA 95008-6557

Prepared By:
Shohet & Kahn PR
2959 S. Winchester Blvd. Campbell, CA 95008
Murry Shohet (408) 379-7434

Contacts:
FORCE USA: Wayne Fischer (408) 370-6300
FAX: (408) 374-1146
FORCE GmbH: James Hole (089) 608140
FAX: (089) 6097793

Distributed Real-Time Multiprocessing on an Inexpensive 68030-Based VMEbus Board

Design Employs Low-Cost Methods to Achieve Fully-Featured Message Broadcasting and 32-Bit Dual Port RAM

CAMPBELL, CA., April 25, 1989 — FORCE COMPUTERS CPU-33 establishes a new performance-to-price mark for 16.7 and 25MHz 32-bit multiprocessing on the VMEbus.

The CPU-33 emphasizes the use of gate arrays plus cost-efficient manufacturing to achieve a pricing structure below competitive products. "We specifically set out to craft a [illegible]-based message-broadcasting engine that exploits technology as well as the most economical manufacturing methods," said Wayne Fischer, Director of Marketing. CPU-33 pricing begins below $2,400.

The CPU-33 offers impressive features to designers of high performance, cost-sensitive real-time simulators, machine and process controllers, robotics and security systems, among many other industrial applications. The CPU-33 features a comprehensive message broadcasting architecture which enables multiprocessing via message exchange between up to 20 CPUs. The board also provides mailbox interrupts and controls 1 MB of on-board shared memory that can be accessed by other processors across the VMEbus. The CPU-33 also offers one parallel and two serial I/O ports; one version includes a socket for an optional floating point coprocessor.

Fischer said the CPU-33 is aimed at industrial OEMs who have been using box-level solutions for multiprocessing. "Given the high performance plus speed selections and I/O capability of the CPU-33, these manufacturers can now build a tailored solution rather than buy a 'compromise in a box'." Fischer repeated a prediction that designs like the CPU-33 are "the leading edge of a new era of design in which VMEbus-based standardized systems displace non-standard custom systems and proprietary solutions."

"In multiprocessing architectures, there is a general assumption that message exchange and mailbox facilities are synonymous with high cost and high performance," Fischer explained. "The CPU-33 is our second design in a series that proves that moderate cost computers can offer all the same architectural benefits. Time-to-market will be much shorter, and risk will be much lower."

Standard Features Are Impressive

The CPU-33's features include:

- 68030 32-bit microprocessor, 16.7 or 25 MHz operation
- 68882 32-bit floating point coprocessor, 16.7 or 25 MHz operation (optional @ 16.7 MHz)
- 1 Mbyte of dynamic RAM (system memory) using 256K X 4 organization
- High-performance 32-bit DMA controller – high speed data transfers locally and across the VMEbus; 32 byte internal FIFO for burst DMA
- Battery-backed 32 Kbyte static RAM
- 2 user EPROM sockets (28- and 32-pin JEDEC) for up to 2 Mbytes of storage
- VME/PLUS technology in FGA-002, a 22,000 gate ASIC which provides message broadcasting, mailbox interrupts and a comprehensive VMEbus interface
- 2-channel FORCE Message Broadcast to up to 20 CPUs simultaneously
- 8 mailbox interrupts (in addition to 7 interrupts allowed at the VMEbus system level)
- Full 32-bit VMEbus interface (A32/24/16, D32/24/16/8); supports unaligned transfers and RMW accesses
- Slot 1 functions include SYSCLK driver, reset generator and support for ACFAIL and SYSFAIL
- 4-level arbiter, with software-selectable arbitration mode; reset timeout is also programmable
- IACK daisy chain driver
- Real time clock with on-board battery backup
- 2 serial channels provided by use of the 68562 Dual Universal Serial Communications Controller (DUSCC). These channels are available on P2 as well as from the front panel
- 2 Parallel Interface/Timers offer a 12-bit parallel port and 2 24-bit timers
- VMEPROM real-time PDOS kernel, monitor and debugger

CPU-33 Software Includes VMEPROM

Enhancing the usefulness of the CPU-33 is VMEPROM, a true real-time operating system kernel that also includes a monitor and debugger. It is installed in EPROM to yield operational capability as soon as the board is inserted in an active backplane. VMEPROM is based on PDOS, a popular operating system from Eyring Research Institute.

The CPU-33 will also be available for use with third-party real-time systems and kernels, including VxWorks, pSOS, VRTX32, OS-9 and PDOS.

Price & Availability

All version of the CPU-33 are available for immediate delivery. Equipped with a 16.7MHz 68030, the CPU-33XN is priced at $2,990 (1-9). A 25MHz version (CPU-33XB) is priced at $3,290.

The CPU-33 is shipped with comprehensive documentation (1200 page manual), including thorough details on the user interface and kernel system calls provided by the VMEPROM package.

About FORCE COMPUTERS

The leading independent designer and manufacturer of VMEbus products, FORCE is now in its seventh year. The company has completed 25 consecutive quarters of profitable operation. Force is headquartered in Campbell, California with subsidiaries in West Germany, France and the United Kingdom. Sales, service and product support are provided on a worldwide basis.

CAREEN 68K brings the power of OS-9/68000 in your IBM-PC

The high performance Add-In-Board CAREEN 68K changes the character of your PC from a simple documentation tool to a powerful real-time software development system.

If you want more informations, here is your contact address:

LP Elektronik GmbH
Ettishofer StraBe 10c
D-7987 Weingarten
West Germany

Telefax: (0751) 53199
Telephone: (0751) 52327

PROFILER

Discover which functions within your C or Assembly programs take the most time to run. An invaluable development tool for anyone worried about program speed.

For information on this product and our other OS-9 software development packages: IMP (the Intelligent Make Program), STIMULUS (Artificial Intelligence Language), PAN UTILITIES and WINDOWS (source code library) contact:-

PAN CONTROLS LIMITED,
Drummore, Doune,
Perthshire,
PK16 6AX, SCOTLAND.
Phone (+44) 78685-261.

ADICON is a consulting and engineering firm providing a range of services, including product specifications, design and development, fabrication of prototypes, software, and complete documentation for OEM's. We specialize in the following MOTOROLA products:

HARDWARE DEVELOPMENT

- 6805, 68HC11, 68000/20
- Single Board Computers
- VMEbus systems

SOFTWARE DEVELOPMENT

- C
- Assembly Language
- UNIX

ADICON CONSULTING AND DESIGN

1123 N. Water St., Milwaukee, WI 53202
414/276-6800

HEWLETT PACKARD
PUBLIC RELATIONS DEPARTMENT
3000 HANOVER STREET
PALO ALTO, CALIFORNIA 94304
PHONE 415-857-1501

HP ANNOUNCES EARLY DEVELOPMENT SUPPORT PLANS
FOR MOTOROLA 68040 MICROPROCESSOR

Hewlett-Packard Company today announced it is designing lifecycle-development tools for the Motorola 68040 microprocessor. HP will provide a real-time, full-speed emulation system, a probe interface to logic analyzers, a cross C language system, a C- and assembly-language debugger, and a software-test system.

HP believes that this toolset will offer designers the most extensive set of integrated tools for embedded microprocessor development available anywhere.

The MC68040 represents a leap in performance for the MC68000 family and at the same time allows customers to reuse their existing MC68000 code. HP development tools already support existing members of the MC68000 family, including the MC68020.

"Industry demands real-time emulation and analysis and optimized software-development tools when designing with 32-bit microprocessors," said David C. Dayton, general manager of HP Logic Systems Division. "To our knowledge, HP is the only vendor today planning to provide all of these tools for the MC68040."

Motorola and HP have exchanged technical information, enabling early development of emulation, analysis and software tools. This arrangement will allow earlier design starts with the MC68040 using HP's quality tools. The availability of these tools will be announced later this year.

The new emulator, which will be part of the HP 64700 series emulators, will provide support for physical and logical memory accesses. Electrical and mechanical interfaces will be optimized for improved plug-in capability. These features will make possible more complete analysis and debugging of prototype designs. The emulator also will have a firmware monitor that will enable the emulator to be hosted by any computer.

HP will market a microprocessor interface to link its HP 1650A and 16500A logic analyzers to the MC68040. The probe interface will provide a complete mechanical and electrical connection with the processor. Users of the MC68040 will be able to track software flow in MC68040 mnemonics, and isolate hardware problems.

The HP 1650A logic analyzer offers 100-MHz timing and 25-MHz state analysis on 80 channels. The HP 16500A logic analyzer can be user-configured as a timing/state analyzer with up to 400 channels of general-purpose logic analysis, a digitizing oscilloscope or a pattern generator.

"Our logic-analysis support will provide a competitive edge to early MC68040 users at minimal cost, because many of them already own our logic analyzers," said Thomas A. Saponas, marketing manager of HP's Colorado Springs Division.

Extensive software support is planned. The Cross C language system will support the proposed ANSI C standard, as well as Motorola assembler mnemonics, enabling users to port their existing code from previous MC68000 designs to MC68040 designs. The compiler system will compile source code into highly space- and time-efficient executable code that takes full advantage of the microprocessor's instruction set and address modes.

• HARDWARE DESIGN, BREADBOARDING, AND PROTOTYPING

• SOFTWARE DESIGN, PROGRAM ASSEMBLY, AND FIRMWARE IMPLEMENTATION

• INTERFACE DESIGN AND DEVELOPMENT

• MICROCOMPUTER BASED MEASUREMENT AND CONTROL

• DESIGN OF SINGLE CHIP MICROCOMPUTER BASED PRODUCTS A SPECIALTY

MICRODYNAMICS
P.O. BOX 2716
WARMINSTER PA 18974 (215)-357 6805

Fulfilling the needs of the nation's high tech community for over a decade.

Leaders in Hardware and Software consulting for real-time military and industrial applications.

AI Microsystems, Inc.
One Naperville Plaza
Naperville, IL 60540
(312) 416-2177

Tomorrow's Technology Today!

RADON

Use Your Computer To Measure Radiation. Write Or Call For Our Sensor Catalog.

For more information please call or write:
LaGrange Instruments, Inc.
Kuchler Drive
LaGrangeville, NY 12540
(914) 223-3336

MOTOROLA INC.

Microprocessor Products Group
6501 William Cannon Drive West
Austin, Texas 78735-8598

For further information contact:

EDITORIAL CONTACT
Nick Stavlak
Cunningham Communication, Inc.
(408) 982-0400

READER CONTACT
Dick Spain
Motorola Inc.
(512) 891-3260

Motorola Announces 68300 Family

New Family Delivers 68020 Workstation Power
with Sophisticated on-Chip Features

AUSTIN, Texas, April 17, 1989 — Motorola's Microprocessor Products Group today announced a line of 32-bit devices targeted at embedded control applications. Based on Motorola's popular 68020 microprocessor, the new 68300 family adds a host of sophisticated on-chip features to bring the power of computer workstations to consumer and industrial control applications. General Motors plans to use the first 68300 family member, the 68332, for future automotive applications.

Motorola's 68020 (020) is the industry standard 32-bit architecture for engineering workstations and multi-user computers. The 68000 family is also the microprocessor of choice for embedded applications, with more than 13 million devices installed in non-computer products such as laser printers, telephone switching systems and robots.

The 68300 family builds on the strengths of the 68020. Its compatibility with the 68000 family gives users immediate access to a worldwide base of 32-bit software and support tools already in existence. The 68300 family also utilizes the functionality standards set by Motorola's highly successful HC05 and HC11 8-bit microcontrollers.

"We see an excellent opportunity to combine our successful microcontroller methodology with the highly regarded 68000 architecture to furnish customers with 32-bit performance for embedded control," said Gary Daniels, general manager and vice president of Motorola's Microcontroller Division. "We are the only company in the world that can extend our architecture from automobiles to high-performance computer workstations."

Architectural Features

Using a common internal communications platform called the intermodule bus, the 68300 line will incorporate a number of intelligent peripherals uniquely adapted to particular markets. Because these units can function autonomously, they greatly decrease Central Processing Unit overhead, thereby dramatically boosting system performance. Applications that can use the 68300 include automobiles, robots, compact disc players, phone systems and a host of computer support needs.

Embedded Control: Powered by Motorola Microcontrollers

While microprocessors are valued for their pure processing power, embedded control applications require much more than conventional performance. These applications demand a microcontroller that can perform a number of operations simultaneously, with low power dissipation and at low cost. The majority of embedded control applications rely on microcontrollers with features such as sophisticated communications circuitry, timers and substantial amounts of memory, all integrated on chip. This increased functionality/decreased part count is essential for high-volume applications such as consumer electronics and household appliances. The 68332, for example, has static design, allowing it to adapt its electrical current requirements and processing speed to meet the changing needs of an application's environment.

Motorola Introduces a Low Cost Evaluation System for First 32-bit Microcontroller

AUSTIN, Texas, April 17, 1989 — Motorola's Microprocessor Products Group today announced a low cost evaluation system for its 32-bit microcontroller, the 68332. The system, called the 68332EVS, enables hardware and software engineers to develop applications and products for the new microcontroller. The 68332EVS is supported by the IBM PC and compatible computers.

The 68332, also introduced today, is the first 32-bit microcontroller. Based on Motorola's 68020 microprocessor, the 68332 combines the power of a technical workstation with the functionality of an entire circuit board on a single chip. A number of consumer and industrial companies have already endorsed the product. General Motors, the world's largest company, has announced plans to utilize the 332 for managing a variety of automotive control functions in its next-generation automobiles.

"The automotive electronics market is the catalyst for the consumer electronics marketplace," said Gary Daniels, general manager and vice president of Motorola's Microcontroller Division. "With GM's plans to use the 68332, the device will spread to other high-volume applications. The development system will allow hardware and software developers to realize the potential for the 32-bit microcontroller market."

Availability

The 332 is available now in beta-sampling. Volume production will begin in the fourth quarter 1989. Sample pricing at introduction is set for under $125. Motorola also has available a complete evaluation system, including hardware evaluation board, software and documentation.

Key Technical Features

- 1 micron HCMOS static design
- 132 Pin PQFP package
- 16.77 Mhz clock at 5 volt supply (300mW typical)
- 2k-bytes of static RAM on-chip
- 32-bit 68020-based CPU with added instructions for controller applications
- Intelligent 16-bit timer with 16 user programmable channels and pins
- Two serial I/O sub-systems
- On-chip programmable chip select logic
- System failure protection
- Fully static design with low power capability

ATARI ST

OS9 Professional™ incl:
C, Basic, Emacs, Kermit,
S record download, OS9 to TOS transfer,
Sculptor™, Dynacalc™, Stylograph™

8 Serial Port board for the MEGA ST.
12 bit Analogue I/O interface.
16 bit Parallel Interface.
EPROM Programmer.

UNISON,
T.J.P. Electronics Ltd.
3 West Street,
Scarborough,
North Yorkshire,
ENGLAND. U.K.
YO11 2QL

Tel U.K. 723-378837
FAX U.K. 723-500435

COMPUTER PERIPHERIALS
FOR THOSE WHO
LIKE TO BUILD
THEIR OWN.
FOR MORE
INFORMATION
SEND A S.A.S.E.
TO

HARLEEN FRANCISCO
8332 PEGGY ST.
TAMPA FL.
33615

XYZ
ELECTRONICS INC.
A NAME YOU'LL REMEMBER
ROUTE 12 BOX 322
INDIANAPOLIS, IN 46236
(317) 335-2128

First to offer OS-9
68000 on the STD Bus.
Full line of memory,
serial, parallel,
analog, digital, and
video I/O. Full
systems too!

The 68332EVS consists of three units. A small microcomputer, called the Business Card Computer (BCC), contains a surface mount 68332, 128 Kbytes of EPROM, 64 Kbytes of RAM and a RS-232 port. The BCC is the core of this low cost development system, enabling stand-alone evaluation of the 68332 on a computer the size of a business card. The BCC development interface (BCCDI) contains an 8-bit 68HC11 microcontroller and a breakpoint chip which monitors address lines and detects hardware breakpoints during emulation. The platform board has sockets for external RAM and is used to connect the BCC to the user's PC.

The 68332EVS can be used in different ways, depending on the stage of product development. For early evaluation of the 68332's features, the BCC is plugged into the platform board, which then connects to the user's PC.

For the initial development of any application or product, the BCC development interface piggybacks onto the BCC and plugs onto the user's target board. By then connecting to the user's PC through the RS-232 port, developers will have the 68332 up and running, eliminating time spent putting together a processor with EPROM and RAM for the first stages of application development. Developers can then emulate the 68332 running on the PC so that hardware and software products and applications development can begin.

For later stages of application and product development, the BCCDI plugs onto the EVS platform board and is hooked by an eight wire cable to the user's target board. The target board, with a surface mount 68332, then allows emulation via the "background mode" of the 68332.

When developers wish to show their initial applications to customers or trade show audiences without traveling with a PC, they can program the application into the EPROM on board the BCC and plug the BCC into a target system. The BCC is so compact, it can fit into a developer's shirt pocket. This configuration enables developers to emulate their applications in development without extensive hardware requirements.

The 68332EVS will be available with general sampling of the 68332 in the fourth quarter 1989. The cost for the entire system is set for $500. The BCC is available as a stand-alone system. It will be available with general sampling in the fourth quarter 1989 with the price set at $332.

Motorola Announces Hardware and Software Development Support for the 68332

Tools Speed Software Development for First 32-bit Microcontroller

AUSTIN, Texas, April 17, 1989 — Seven companies today announced extensive software development support for Motorola's new 68332 (332) microcontroller. Companies introducing products are Hewlett-Packard, Intermetrics, Introl Corp., Ready Systems, Software Components Group and Tektronix. They are providing various development tools, including operating systems, emulators, assemblers, debuggers and compilers. Motorola has also stated that their MDS-300 development system will be available for 332 application development.

The 332, introduced today, is the world's first 32-bit microcontroller. It is a combination of Motorola's successful 68000 microprocessor family and on-chip peripherals, delivering high performance at low cost. General Motors, the world's largest company, made public its intentions to use the chip in future automobiles.

"With General Motors' plans to use the 332, we know our product is positioned to become the industry leader," said Gary Daniels, vice president and general manager of Motorola's Microcontroller Division. "These software and hardware tools are integral to our strategy to broaden the market for 32-bit microcontrollers."

Microcontrollers are the workhorses of the electronics industry, with over $2.3 billion sold in 1988. These semiconductors are also the brains of a multitude of products in the home and office. Microwave ovens, pagers, automobiles, elevators and TVs all depend on microcontrollers for their processing power.

Typically, microcontrollers have been 4- or 8-bit devices that require little software development support. Traditionally, 4- and 8-bit device programs are written in the language of the controller. However, with 32-bit products, software development is crucial. Because the programs designed for these chips are large and are written in a language other than that of the microcontroller, a number of tools, including compilers, assemblers and emulators, are needed to ease software development. These seven companies' products provide all the necessary tools to develop software for Motorola's sophisticated microcontroller.

Hewlett-Packard's Logic Analyzer Division (Boulder, Colo.) has announced a real-time, 16MHz emulation system, a cross C language system, C language and assembly language debuggers and a software test system. These tools will be available in fourth quarter 1989.

Intermetrics Corp. (Cambridge, Mass.) is developing optimizing C compilers, macro assemblers and XDB source level debuggers supporting several popular debugging environments. The products will be available in late 1989.

Introl Corp. (Milwaukee, Wis.) has developed a C language cross-compiler system, a Modula-2 language cross-compiler system, an assembly language development system and a source level debugger. Introl's 332 software development products will be available in June 1989.

Motorola's Development Systems group (Austin, Texas) has announced the MDS-300 development system for the 332. The MDS-300 system consists of a host computer, C compiler, assembler/linker and source level debugger, a development station, a target system, an emulator module and an analyzer.

Ready Systems (Sunnyvale, Calif.) has introduced a real-time development system. The system includes a multi-tasking kernel, an advanced real-time debugger and system monitor, a real-time C development tool package and Ready's front-end analysis and design tools. Ready Systems' 332 development tools are available now.

Software Components Group (San Jose, Calif.) is porting pSOS+, its family of real-time operating system components, to the 332. pSOS+ for the 332 will be available in the second quarter of 1989.

Tektronix (Broomfield, Colo.) has announced the 1230 Logic Analyzer for the 332, providing disassembly support for debugging and optimizing 332 systems. A complete 332 disassembly system, including the 1230 Logic Analyzer, will be available in late 1989.

Logic Analyzer Division
P.O. Box 12132
Portland, OR 97212

TEKTRONIX 1230 LOGIC ANALYZER SUPPORT FOR MOTOROLA'S NEW 68332 32-BIT MICROCONTROLLER

Beaverton, OR — Tektronix, Inc. today announced the 1230 Logic Analyzer support of Motorola's new 68332 32-bit microcontroller. Tek is the first company to provide disassembly capability for the 68332 chip.

The 68332, also announced today, is the world's first 32-bit microcontroller. It is a combination of Motorola's successful 68000 microprocessor family and on-chip peripherals, delivering high performance at a low cost.

The Tektronix 68332 disassembly probe provides instrument setup and disassembly post-processing for debugging and optimizing 68332 systems. Designers of 68332 systems can use the 68332 disassembly probe to monitor their system's activity in real-time without the burden of added wait states and debug code that can compromise real-time code execution.

DOCTOR DESIGN
68XX
and
68XXX
WIZARDS

LET US DESIGN YOUR HARDWARE AND SOFTWARE APPLICATIONS

- Embedded Applications
- Single Board Computers
- VME and Multibus

DOCTOR DESIGN, INC.
5415 Oberlin Drive
San Diego, CA 92121
(619) 457- 4545
FAX : (619) 457-1168

SCADA SYSTEM/CONSULTING

The Real-Time Control System package RCS-7 provides the basic functions of a SCADA (Supervisory Control and Data Acquisition) system. RCS-7 utilizes multi-tasking and distributed systems and supports most RTUs and programmable controllers. RCS-7 is field-proven in the OIL/GAS, ELECTRIC UTILITY, WATER/WASTE WATER and FACTORY AUTOMATION industries. The RCS-7 system and expert consulting services are available. Please contact:

AUSPEX INC.
3730 KIRBY DR., STE. 650
HOUSTON, TX 77098
TEL:(713)524-3100
Fax:(713)524-8045

A

OS-9/68020, 68000, 6809
C, Forth, Asm, Sculptor
Embedded Systems
Device Drivers
Applications

RICHARD HOGG

ACE PROGRAMMER

(717) 343-1317
430 S. Edwards Ct.
Scranton, PA 18504

A

Quick and Convenient Connections. The 68332 disassembly probe connects to a 64-channel 1230 Logic Analyzer by use of a universal preprocessor, the 1Z300PA (disassembly probe adapter). The 68332 disassembly probe connects to the system under test with a low-profile, low-impedance, 132 pin PQFP (plastic quad flat pack) socket adapter. Because of the quick-disconnect mechanism between the disassembly probe and the 1Z300PA preprocessor, it takes only seconds to reconfigure the 1230 to support another microprocessor.

For additional information on the 1230 Logic Analyzer, and its military-approved partner—the 1230B Logic Analyzer, contact Tektronix, Inc., Logic Analyzer Division, P.O. Box 1210-32, Portland, Oregon, 97212. Or call toll free 1-800-245-2008.

ntrol Corp. Announces Software Development Products for Motorola 68332 Microcontroller

Milwaukee, Wis.—April 17, 1989—Introl Corp. today announced four new software development products supporting the new Motorola 68332 32-bit microcontroller. Introl's new products include a C language cross-compiler system, a Modula-2 language cross-compiler system, an assembly language development system and a source level debugger.

The Motorola 68332 is the first full 32-bit microcontroller that integrates the workstation-level performance of Motorola's 68020 microprocessor with the on-chip peripherals of the popular 68HC11 8-bit microcontroller.

"The Introl software development tools for the 68332 are an exciting addition to our current line of support tools for the 6800 and 68000 microprocessor families," said John Wisialowski, president and chief executive officer of Introl. "We see a vast and untapped market for the 332 because of its 32-bit performance and compatibility with the 68020 microprocessor."

The Introl-C/68332 cross-compiler system is a set of programming tools that fully supports both C language and assembly language software development for the 332. The cross-compiler system combines ease of use, powerful and versatile ROM-support capabilities and excellent output code efficiency, greatly facilitating the development of compact, fast executing embedded applications software for stand alone ROM-based 332 environments.

The Introl Modula-2/68332 cross compiler system is a set of programming tools that provides both Modula-2 language and assembly language software development support for the 332. The Modula-2 compiler fully supports the language as defined by Niklaus Wirth, as well as supporting a number of language extensions which further enhance its suitability for embedded systems programming.

The Introl 68332 assembly language development package provides the embedded systems developer with a powerful assembler and a set of support utilities for efficiently transforming 332 assembly language programs into ROMable output code. The relocating macro cross-assembler supports conditional assembly, macro substitutions, file inclusion, listing control, symbol cross-reference listings and the full instruction set and all addressing modes of the 332.

The Introl 68332 source level debugger is a powerful tool for debugging C language and Modula-2 language programs. With the Introl debugger, software developers can selectively start, stop and monitor the execution of embedded 332 programs using the same high-level language terms, definitions and structures employed in the original source program.

These Introl 68332 software development products are available for a wide variety of host development systems including IBM PC, XT, AT and clones, Macintosh, VAX and Micro VAX, Sun, HP9000, Apollo and many others.

Introl's full line of 68332 software development products will become available June 1, 1989. Prices for the Introl-C/68332 and the Introl Modula-2/68332 compiler systems start at $2,000. Prices for assembler packages and debugger packages start at $1,000.

Contacts:
Introl Corp., John Wisialowski, (414) 276-2937
Motorola Inc., Dick Spilo, (512) 891-3260

Nationwide Specialists In The Recruitment And Placement of Software Engineers

Compiler Design, Kernel Modification, OS Internals Voice and Data Communication, Network Design, Windows, X Windows, NeWS, Desk Top Publishing, CAD/CAE/CAM/CASE, Language Design, OOP, AI Graphics, Raster Technologies, Laser/Disk/Video Systems Database Design, Porting Systems Architecture "Start Up", Venture Capital Emerging Technology Companies

Nayland Associates
Route 2, Box 352
Nebo, NC 28761
(704) 652-1801

Intermetrics

EDITORIAL CONTACT:
Allen Ready — Intermetrics, Inc. (617) 661-1840
Dick Spilo — Motorola, Inc. (512) 891-3260

Intermetrics to Add 68332 Development Software

Cambridge, Mass. -- Intermetrics announced today that the company will offer a full set of development software for Motorola, Inc.'s 68332 microcontroller. The new products will include optimizing C compilers, macro assemblers, and XDB source level debuggers supporting several popular debugging environments.

William E. Carlson, General Manager of Intermetrics' Development Systems Group, said "We look forward to extending our strong support for the Motorola microcontroller family. We see the 68332 as a high-performance addition to Motorola's leading microcontroller offerings". Intermetrics supports Motorola controllers such as the 6800 and 68HC11, and will be offering the 68332 tools as part of its overall focus on software for embedded processors.

The 68332 is the first 32-bit microcontroller. It integrates the proven performance of Motorola's 68020 microprocessor with the on-chip peripherals of the popular 68HC11 8-bit microcontroller.

The compilers will be fully compatible with ANSI and Kernighan and Ritchie C standards, and will produce highly optimized object code for the 68302 processor. The optimizations performed using global flow analysis will include common sub-expression elimination, register allocation, and loop rotation. The optimizer reduces the size of generated code by 10%-50% and leads to substantially faster run-time performance.

The assemblers will be 100% compatible with the Motorola assembly language standards for the 68332, and will feature full macro capabilities. The toolset will also include utilities designed expressly for embedded systems development. These include a Linker/Locator, ROM processor, and a Formatter that produces all popular industry formats, including HP 64000.

Intermetrics will also interface its XDB source level debugger to 68332 instrumentation, providing support for true source level debugging. XDB features a multi-window user interface. Besides viewing source code and XDB commands, users can summon windows displaying registers, monitored variables and expressions, a stack trace, simulated I/O, and a C statement execution trace. There are also pop-up windows to view the active breakpoints, User Defined Functions (UDFs), and the multi-tiered help facility.

Public Relations Contact
Jim Ready (Ready Systems) 408/736-2600
Karen Barnaba (Ready Systems)
Lisa Pigliotti (Pranam & Assoc.) 408/453-5220

Ready Systems Offers Total Real-Time Solution for Motorola 68332 Microprocessor

Sunnyvale, Calif., April 17, 1989 -- Ready Systems today announced the first comprehensive, fully integrated, real-time development solution available for the Motorola 68332 microprocessor. The package includes a real-time multitasking kernel (VRTX32), an advanced real-time debugger and system emulator (RTScope), a real-time C development tool package (RTC), and Ready's front-end analysis and design tools (CARDtools).

The Motorola 68332 marks a major advance in the development of highly integrated microcontrollers. By combining an enhanced instruction set upwardly compatible with the Motorola 68000 series of microprocessors, faster clock speeds, and powerful on-chip peripheral support and intelligence, the 68332 provides a single chip engine ideal for embedded control applications. General Motors Corporation has already announced their intention to use the

According to Jim Ready, executive vice-president of Ready Systems, "Today, Ready Systems is able to offer the same fully integrated development and run-time software solution for the 68332 that has been so successful for other members of the Motorola 68000 family of microprocessors. We are pleased to once again extend our long standing relationship with Motorola. The combination of our highly integrated software tools and Motorola's integrated chip means greater design efficiency for less cost."

Ready Systems offers a fully integrated solution that addresses every phase of the software lifecycle, from front-end analysis and design tools, to reusable runtime software components, to documentation and maintenance support. The integrated approach significantly reduces the complexity of real-time embedded systems software engineering, improving cost of development and time to market.

Ready's front-end design tools, CARDtools, is the first CASE product that supports multitasking real-time embedded systems design. CARDtools integrate front-end design tools with the knowledge of the target operating system and allow for real-time performance verification against system timing requirements. Using CARDtools, software coding and maintenance costs are significantly reduced by automating requirements analysis, design modeling and performance analysis, with output support for object-oriented design, program design language (PDL), H/S S/W interface specification, published DoD-STD-2167 documentation generation, and open systems access.

Once design is completed using CARDtools, the code can then be compiled, debugged, and downloaded to the target with RTC, a real-time development environment including a C language compiler, and RTscope, a multi-tasking system level debugger and system monitor. The application is then efficiently executed by Ready's real-time kernel, VRTX32, a high-performance multitasking executive with an advanced internal design. VRTX32 ensures deterministic response through fixed-cost system calls independent of variables such as tasks, queues, interrupts and system overhead.

"Software support is essential for a new microprocessor to be effective," explained Brian Wilkie, midrange operations manager of the Microcontroller Division at Motorola. "The superior software solution provided by Ready Systems, combined with our strong hardware environment, will provide an unbeatable solution for embedded systems customers in many technical system market segments," he added.

Ready Systems provides board support and peripheral driver software for the 68332 through its Applications Group. This group, with in-depth experience on the 68332, will now support embedded designs targeting this processor.

Contact: Steve Jobe
Software Components Group, Inc.
(408) 437-0700

SOFTWARE COMPONENTS GROUP ANNOUNCES REAL-TIME EMBEDDED OPERATING SYSTEM SOFTWARE FOR MOTOROLA'S 68332

San Jose, April 17, 1989 - Software Components Group, Inc. (San Jose, CA), a leading supplier of real-time microprocessor software, announced that it is porting its popular pSOS$^+$ family of real-time operating system components to the Motorola 68332 high-end embedded microcontroller architecture.

The pSOS$^+$ family provides a complete, leading-edge solution for high-end real-time embedded system designs. The products have been widely used in such diverse application areas as telecommunications, military electronics, medical instrumentation and factory automation. On 68000/68020 processors alone, pSOS has logged well over 1000 major design wins and is considered by many as the industry standard.

"We see the 68332 as a landmark development for the micro-controller market", according to Steve Jobe, Manager of Strategic Marketing. "It will be a great success with many of our high-volume pSOS customers. Not only does it offer 32-bit performance, high functional integration, and reduced cost, but, most significant of all, software compatibility with the 68000 family. A user can port an existing pSOS-based application, or start a new one with minimum effort and zero learning curve, literally."

Release of the pSOS$^+$/332 Real-time Kernel and pROBE$^+$/332 System Debugger is planned for the Second Quarter of 1989. Release of the company's complete set of support software for the 68332 will be made by the Fourth Quarter of 1989.

Software Components Group, Inc. is headquartered in San Jose, California. The pSOS$^+$ family of real-time embedded software components includes the pSOS$^+$ Single and Multiprocessor Multitasking Kernel, pROBE$^+$ System Debugger/Analyzer, XRAY$^+$ Remote Source Level Debugger and pHILE$^+$ File System Manager. By using these standard, robust building blocks in products or systems with embedded microprocessors and microcontrollers, design engineers achieve higher reliability, performance and significantly reduce development and maintenance costs, and time-to-market.

Editorial Contacts:

Clelia deMoraes
(719) 590-5889

John Marshall
(719) 590-5985

HP ANNOUNCES EARLY DEVELOPMENT SUPPORT PLANS FOR MOTOROLA 68332 MICROCONTROLLER

PALO ALTO, Calif., April 17, 1989 -- Hewlett-Packard Company today announced it is developing lifecycle support for the Motorola 68332, the world's first 32-bit microcontroller.

HP will provide a real-time, 16-MHz emulation system, a cross c language system, C- and assembly-language debug system, and software-test system. This toolset will offer designers the most extensive set of integrated tools for the microcontroller available anywhere, HP said.

The 68332 provides significant performance gains over existing 8-bit and 16-bit microcontrollers. "We expect this microcontroller to be popular because it offers 32-bit performance and is still code-compatible with the popular 68000 family," said David C. Dayton, general manager of HP's Logic Systems Division.

Motorola and HP have exchanged technical information enabling early development of emulation, analysis and software tools. This arrangement will allow earlier 68332-based design starts using HP's quality tools. HP expects to have its emulator in beta test by late summer. The emulator and a full set of software tools are expected to be available later in the year.

WINDRUSH
Micro Systems Ltd.

Station Road (Reg. Office)
Worstead, North Walsham,
Norfolk NR28 9SA England
Telex: 975048 WMICRO G
Tel: (0692) 404086
Fax: (0692) 404091

WINDRUSH HARDWARE & SOFTWARE RESCUE SERVICE

Windrush Micro Systems Limited announces the immediate availability of access to a team of Motorola MC6809, MC68000 and MC68020 design specialists.

Design engineers are now facing formidible learning curves in the design and programming of systems using the latest generation of 16-bit and 32-bit microprocessors. This often results in projects taking two to three times the original project estimates.

Windrush offer a full range of services ranging from project start-up assistance to complete design, prototyping, programming and manufacture.

Windrush have specialised in the Motorola and Hitachi 68XXx family since the introduction of the MC6800 in 1975. The company has not diluted its expertise by involvement with the Intel, Rockwell or Zilog (et al) families of processors. Our expertise ranges from the application of single chip micros, eg 6801, 6803, 6805 and 6811 to the latest generation of 32 bit processors, eg 68020 and 68030.

The company also specialises in the application of microprocessors in microprocessor development systems using FLEX and OS-9/68k, process/machine tool control systems and data logging equipment.

We *always* offer fixed cost quotes. There are *never* any hidden costs. You know, in *advance*, exactly how much the project is going to cost, right down to the last penny.

Before you start an in-house project why not give us a call ... you may find that we can do the job more economically and in a more timely manner. Important factors if you want to beat your competition.

For further information contact Bill Dickinson on (0692) 404086

GESPAC Inc.
50 West Hoover Ave.
Mesa, Arizona 85210
Tel. (602) 962-5559
Fax. (602) 962-5750

Reader Contact: Don Bizioa
Editorial Contact: Coama Pabouctaidia

CONTROLLER BOARD AND TCP/IP SOFTWARE
CONNECTS G-64 SYSTEMS TO ETHERNET

MESA, AZ, April 25, 1989--In an effort to allow G-64 based systems to communicate with existing computer resources, GESPAC is announcing Ethernet communications hardware together with TCP/IP protocol support for the OS-9 real-time operating system. The EVLAN-11 board allows connection of a G-64 based system to any IEEE 802.3 LAN through the appropriate network transceiver, connecting to Cheapernet or Ethernet cabling on new or existing broadband or baseband networks. There is 64K local memory available to the controller for buffering, plus both the physical and data link layers are handled in hardware in order to free the host processor from much of the communications overhead. Full generation and interpretation of the packet preamble, plus CSMA/CD arbitration is handled by the on-board processor.

These features make this controller an efficient peripheral, while providing real-time systems with an unprecedented level of communication flexibility.

MATRIX Corporation
1203 New Hope Road
Raleigh, NC 27610

Contact: Mary Klang
Phone: (919)833-2000
FAX: (919)833-2550

COMBINATION DATA ACQUISITION BOARD IS A COST-EFFECTIVE ALTERNATIVE TO MULTIBOARD SOLUTIONS

RALEIGH, NC--MATRIX Corporation announces the MD-DAADIO, a high-performance, combination D/A, A/D, and PIO board. The single-board MD-DAADIO incorporates the collective functionality of three stand-alone I/O boards and thus represents a highly-integrated, space-efficient, and cost-effective solution for data acquisition applications.

The Analog-to-Digital section of the MD-DAADIO offers more functionality than is typically available on a dedicated A/D board. A unique pipelined A/D conversion method developed by MATRIX engineers, gives the MD-DAADIO an effective sampling rate of 125 KHz. In addition, the MD-DAADIO offers a single-cycle conversion technique that automatically starts the next conversion upon reading current conversion data. The MD-DAADIO also features software selectable gain values of 1, 2, 4, 8, 16 (or alternatively 10, 20, 40, 80, 160). These gain values support the wide-ranging functional requirements of today's A/D applications. Furthermore, the MD-DAADIO supports external triggering and thus is well-suited for externally-clocked synchronous applications.

The 12-bit Digital-to-Analog section of the MD-DAADIO provides up to 8 channels of analog output. These channels are configurable as either all voltage, all current loop, or some combination of voltage and current loop. The signal characteristics of each output channel can be independently set for voltage range and mode (unipolar/bipolar). In addition, multiplying D/A capability allows output scaling to external AC or DC references. Each channel interprets digital input as straight binary (unipolar), offset binary (bipolar), or twos complement (bipolar) depending upon the particular jumper setting of that channel. This alleviates the need to complement or offset D/A data in software.

The PIO section of the MD-DAADIO features 48 lines of parallel I/O and interleaved grounds. All six of the board's 8-bit I/O ports are individually software configurable for data directionality. Furthermore, the first port is capable of interrupting the VMEbus on change-of-state or byte recognition and is bit-maskable. All I/O ports support high current operation and are capable of sinking up to 24 mA and sourcing 15 mA. Additionally, both 50-pin headers provide 5V power for interfacing to OPTO-22 type modules.

Gateway

Contact: Ronna Alsnuck
Gateway Design
Automation Corporation
Two Lowell Research Center Drive
Suite 300
Lowell, MA 01852
(508) 458-1900

Design Methodology Fact Sheet
Motorola Microprocessor
Products Division
April 13, 1989

NEW 68332 CHIP DESIGNED WITH VERILOG-XL®

Design Complexity:	• 422,000 transistors • 5 functional modules in one chip • 1 micron HCMOS design • embedded 32-bit 68020 complexity CPU • multiple on-chip peripheral controllers • microcoded controllers
Methodology:	• top-down design using Gateway HOL • mixed-level simulation and verification running on workstations • microcode verification prior to gate level implementation
Reasons for Choosing Verilog-XL:	• speed • accuracy • mixed-level environment
Results:	• dramatically decreased time-to-market (verified microcode a year ahead of schedule)

###

The Australian developed and manufactured 68IPU30 multipurpose industrial processing unit, based on 8MHz Motorola MC6809 microprocessor, is directed at the OEM and educational markets as a high level building block for control and monitoring. In conjunction with a personal computer it also may be used as an intelligent data logger. Both hardware and software syeteas are designed to provide aignal conditioning, coaputational functions , self checking and communications capability. Drivers for Liquid Displays and parallel/serial printers aa well as RS-232 or RS-422 serial communications capability are available. On board interfaces cater for up to 24 channels of analog input and up to 8 channels of analog output, dual channel serial interface and 32 bits of digital IO and three counter/timers. Analog signal conditioning includes passive filtering or galvanic isolation of up to 300VAC on 12 of the the inputs and software filtering using normal and moving point averages. Four of the analog outputs may be configured for 4-20 Ma operation. An on-board / off-board temperature reference is supported for cold junction compensation when thermocouples are used. On-board aemory includes RAM, EPROM, optional battery back-up RAM as well es a real time clock.

For large data logging applications meaory may be expanded to 2 Mbytes of battery backed CMOS RAM using 68MBR32 expansion cards.

Microgear provides a comprehensive range of software and hardware engineering services. The software is based on a multi tasking re-entrant scheduling and time tabling kernel. Software drivers for this card include assembly and high level language routines written in the Omegasoft Pascal environaent.

Microgear
P.O.Box 154
Churchill 3842
Victoria
Australia
Tel: 051 22 11 57

BIT 3
COMPUTER CORPORATION

8120 Penn Avenue South
Minneapolis, Minnesota 55431-1393
(612) 881-6955 Telex 4997016 BITTHRE
FAX: (612) 881-9674

IBM PS/2-VME ADAPTOR

INTRODUCTION-- The Bit 3 model 443 IBM PS/2-VME ADAPTOR allows the IBM PS/2 to be used as a processor on the VMEbus. The PS/2 behaves just like any other VMEbus processor card. The IBM PS/2 can be the only processor on the VMEbus or it can be a coprocessor. No interim software or software drivers are required to use the ADAPTOR because of Memory Mapping, a technique that allows the IBM PS/2 to address VME memory as though it were PS/2 memory.

An optional Dual Port RAM module provides a common memory directly accessible on both the PS/2 and the VMEbus. Accessing the memory from either bus does not use bandwidth on the other bus. Random access reads or writes to this memory are made by either system as though the memory were local.

The 443 IBM PS/2-VME ADAPTOR consists of two printed circuit cards and a manual. Optional Dual Port RAM plug-in cards are available in 32K, 128K and 1 Mbyte sizes.

SOFTWARE DEVELOPMENT SYSTEMS, INC
4248 BELLE AIRE LANE
DOWNERS GROVE, ILLINOIS 60515 USA
Phone: 1-800-448-7733 or 1-312-971-8170
Fax: 1-312-971-8513

The Assembler

1. It's Fully Featured

The CrossCode C assembler has all the features you'd expect in a professional quality standalone 68000 assembler. It can generate program listings, symbol listings and cross references, and it supports a powerful macro language.

You can *include* source files from within other files, and *includes* are nestable. An OFFSET directive allows you to define assembly language structures. Conditional assembly is provided via nestable IF constructs. And of course there's a complete family of directives to let you reserve both initialized and uninitialized memory space.

Lets You Write Modular Code

You can write your code in a fully modular fashion because the CrossCode C assembler produces relocatable object files. You may combine any number of these files into a final load using the CrossCode C linker. Code and data can be placed into named memory sections (up to 254 permitted) for easy placement at link time.

Supports Advanced Macros

When defining macros, you may use both positional and named macro parameters, and you can perform powerful string manipulations on those parameters. It's also easy to generate unique symbol names in each macro invocation.

Generates Absolute Listings

When you're debugging, you really need to see absolute addresses and fully linked object code on your assembly language listings. But the assembler cannot know where you will eventually locate things at link time, so it generates listings with relative addresses and unlinked object code.

Supports the 68010, 68020 and 68881

You may choose to develop code for the 68000, 68010, 68020, or the 68020 and 68881 together. Via a command line option, you determine whether the assembler will accept the additional instructions and operand modes.

No Limitations

No matter how large your program is, the CrossCode C assembler will assemble it. The assembler imposes no limits on the number of symbols in your program or the size of your input file. Symbol names may contain up to 64 significant characters, and large symbol tables overflow gracefully to disk.

It's Tailor-Made

The CrossCode C assembler gives you maximum flexibility in assigning your code and data into ROM and RAM. With its 254 memory sections, powerful macros, advanced listing capabilities, and special features for C and Motorola compatibility, you'll have all the low-level control you need to build your ROMable application.

Continued from page 47 April 89 issue
Available on Reader Service Disk #34 - See page 62 this issue

FLEX 6809 Diagnostics Disk Drive Test

ROM Test, RAM Test By: Emery Korpi

```
        NOP
        NOP
        NOP
        NOP
        NOP
T1B3    LDA     8,Y     SU STATUS
        BITA    #$01    BUSY?
        BNE     T1BOK   YES
        DEC     T1B2+1  BRANCH
        BEQ     T1BERR  TIMEOUT
        INC     CNTR
        LDX     #$0000  SET RESTORE
        JSR     RESLP   WAIT NOT BUSY
        BRA     T1B1    LOOP
T1BERR  LOX     #GTHN   TIMEOUT
        BRA     T1B4
T1BOK   LDX     #NUL    IN RANGE
T1B4    LDA     #$03    ND OF BYTES
        JSR     XFER    UPDATE MSG
        LDB     CNTR    TIME
        LDA     #$00    TIME MSB
        STD     COUNT+1 STORE TIME
        JSR     CLKCMP  CLK COMPENS
        LDD     #$0000  LO=0SEC
        STD     LIMT
        LDD     #$0022  HI=34USEC
        JSR     PASFAL  LIM OK?
        JSR     UPDMSG  UPDATE MSG
        LDX     #$0000  CLEAR
        STX     T1BMSH+3 LEADING 0'S
        STX     T1BMSH+5
        LDX     #T1BMSG BUSY DEL TIME
        JSR     TESTOK  PRINT
T1C     LDU     #T1CMSH HEAD LOAD DELAY
        LDD     #$2026  TIME D5=HI
        JSR     TIMER   MEAS TIME
        JSR     CLKCMP  CLK COMPEN
        LDD     #$0000  LO=0 MSEC
        STD     LIMT
        LDD     #$4E20  HI=200MSEC
        JSR     PASFAL  LIM OK?
        JSR     UPDMSG  UPDATE MSG
        LDX     #T1CMSG HDLD DELY TIME
        JSR     TESTOK  PRINT
T1D     LDU     #T1DMSH WR PROT STATUS
        LDA     #$0F    RESTORE
        STA     8,Y     CMD REG
        JSR     DELAY
        LDB     8,Y     STATUS
        BITB    #$40    CHECK D6
        BEQ     T1D1
        LDA     #'Y     WR PROT
        BRA     T1D2    CONTINUE
T1D1    LDA     #'N     NO WR PROT
T1D2    STA     ,U      UPDATE MSG
        LDX     #T1DMSG WR PROT =
        JSR     PSTRNG  PRINT
T1E     LDU     #T1EMSH INDEX TOGGLE?
        LDX     #$8000  CLR TIMER
        CLR     TEMP
        CLR     CNTR    CHANGE COUNT
T1E1    MUL             DELAY
        LDA     8,Y     STATUS
        ANDA    #$02    INDEX BIT
        CMPA    TEMP    CHANGE?
        BEQ     T1E2    NO
        INC     CNTR    YES
        STA     TEMP    STA INDEX BIT
T1E2    LEAX    -1,X    DEC TIMER
        BNE     T1E1    LOOP
        LDA     CNTR
        LSRA            GET PULSE CNT
        PSHA            TEMP STORE
        JSR     HEXCON  UPDATE MSG
        PULS    A       PULSECNT
        CMPA    #$02    GT 2 PULSES
        BHI     T1EOK   OK
        INC     ERRFLG
T1EOK   LDX     #T1EMSG INDEX CNT =
        JSR     TESTOK  PRINT
*       TEST 2 - RPM TEST
T2      LDX     #T2MSG  RDTATION SPEED
        JSR     PSTRNG  PRINT
        LDX     #T2CLR  RETEST ADDR
        STX     RETRY   STORE
T2CLR   LDX     #$0000  CLR CNTR
T2WAIT  LDB     8,Y     STATUS
        BITB    #$02    INDEX
        BEQ     T2WAIT  WAIT
CNTHI   LEAX    1,X     INC CNTR
        BRN     T2      DELAY
        NOP
        LDB     8,Y     STATUS
        BITB    #$02    INDEX
        BNE     CNTHI   LOOP
        STX     TIME    INDEX PULSE WDTH
CNTLO   LEAX    1,X     INC CNTR
        BRN     T2      DELAY
        NOP
        LDB     8,Y     STATUS
        BITB    #$02    INDEX
        BEQ     CNTLO   LOOP
        LDU     #T2AMSH MSG POINTER
        STX     COUNT+1 INIT COUNT
        JSR     CLKCMP  CLK COMPENS
        LDD     #$4DBC  LO=199MSEC
        STD     LIMT
        LDD     #$4E84  HI=201MSEC
        JSR     PASFAL  LIM OK?
        JSR     UPDMSG
        LDX     #T2AMSG GET MSG
        JSR     TESTOK  PRINT
*       TEST 3 - INDEX PULSE WIDTH
T3      LDX     #T3MSG  IND PULSE WIDTH
        JSR     PSTRNG  PRINT
        LDU     #T3AMSH GET MSG PNTR
        LDX     TIME    GET PULSE WIDTH
        STX     COUNT+1 PUT IN BIN BUFF
        JSR     CLKCMP  CLOCK COMPENS
        LDD     #$00FA  LO=2.5MSEC
        STD     LIMT
        LDD     #$0226  HI=5.5MSEC
        JSR     PASFAL  LIM OK?
        JSR     UPDMSG  PUT 0 IN MSG
        LDX     #0000   CLR FIRST 2 0'S
        STX     T3AMSH
        LDX     #T3AMSG GET MSG
        JSR     TESTOK  PRINT
*       TEST 4 - TRACK 0 SWITCH TEST
T4      LDX     #T4S    RETRY ADDR
        STX     RETRY
T4S     CLR     CNTR    LOOP CNT
        LDX     #$0000  CLEAR MEM
        STX     DVD+1   DIVIDEND
        STX     DVR+1   DIVISDR
T4S1    BRA     T4A0    NOP/ON PRNT
T4A0    LDX     #T4MSG  TRK 0 TST
        JSR     PSTRNG  PRNT
T4TOG   JSR     RESTOR
        LDA     #$00
        STA     9,Y     TRK REG
        LDX     #$0020  SET TIMER
T4A1    LDA     #$5F    STEP IN
        STA     8,Y     CMDREG
        JSR     DELAY
T4AL1   LEAX    -1,X    TIME
```

```
        BEQ   T4AOUT
T4A2    JSR   DELTS    DELAY
        LDB   8,Y      STATUS
        BITB  #$04     TRKD S/B 0
        BNE   T4A3     NO?
        INC   CNTR
T4A3    LDA   #$0F     RESTDRE
        STA   8,Y      CMDREG
        JSR   DELAY
T4AL2   LEAX  -1,X     TIME
        BEQ   T4AOUT
T4A4    JSR   DELTS    DELAY
        LDB   8,Y      STATUS
        BITB  #$04     TRKD S/B 1
        BEQ   T4A5     NO?
        INC   CNTR     SWITCH CNT
T4A5    BRA   T4A1
T4AOUT  LDA   CNTR     LIMITS
        CMPA  #$05
        BHI   T4B
        INC   ERRFLG   ERRDR
        LDX   #T4AERR
        JSR   PSTRNG   PRINT
        JSR   DPTION
        LBRA  T5       SKIP T4B
T4B     LDA   #$08     LOOP CNT
        STA   CNTR
        JSR   RESTOR
T4B1    LDU   #T4AM3   MSG PNTR
        LDA   #$5F     STEP IN
        STA   TCMD+1   PUT IN TIMER
        LDD   #$0427   TIME D4=LQ
        JSR   TIMER    MEAS TIME
        LDD   DVD+1 (T1)
        ADDD  COUNT+1 SUM RESULTS
        STD   DVD+1
T4B2    LDU   #T4BM3   MSG POINTER
        LDA   #$DF     RESTORE
        STA   TCMD+1
        LDD   #$0426   TIME D4=HI
        JSR   TIMER    MEAS TIME
        LDD   DVR+1    (T2)
        ADDD  COUNT+1 SUM RESULTS
        STD   DVR+1
        CLR   9,Y
        DEC   CNTR     LOOP?
        BNE   T4B1
T4BOUT  LDA   #$03     DIVIDE CNT
        STA   CNTR
T4SHFT  LDD   DVD+1    (T1)
        LSRA           T1:=T1/2
        RDRB
        STD   DVD+1
        LDD   DVR+1    (T2)
        LSRA           T2:=T2/2
        RDRB
        STD   DVR+1
        DEC   CNTR
        BMI   T4B3     LOOP GT 3
        BNE   T4SHFT   LOOP LT 3
        LDU   #T4AM3+3 MSG POINTER
        LDX   DVD+1    (T1)
        STX   COUNT+1 T1/8
        JSR   CLKCMP   CLOCK COMPEN
        JSR   UPDMSG
        LDU   #T4BM3+3 MSG POINTER
        LDX   DVR+1     GET  T2
        STX   COUNT+1   T2/8
        JSR   CLKCMP    CLOCK COMPEN
        JSR   UPDMSG
        LDD   DVD+1    (T1)
        ADDD  DVR+1    T1+T2
        STD   DVR+1
        STD   STPSIZ   STEP TIME
        CLR   DVD
        CLR   DVR
*...
        LDA   #$64
        STA   CNTR
T4B3    JSR   MUL3X1
        LDX   COUNT
        STX   DVD
        LDB   COUNT+2
        STB   DVD+2
        LDA   #$03
        STA   LENGTH
        JSR   MBDIV
        LDX   DVD
        STX   COUNT
        LDB   DVD+2
        STB   COUNT+2
        STB   TEMP     STEP PNT
        LDU   #T4CM2   MSG PNTR
        LDD   #$0005   LO=5%
        STD   LIMT
        LDD   #$005F   HI=95%
        JSR   PASFAL   LIM OK?
        JSR   UPDMSG
        LDX   #$0000   CLR LEADING D'S
        STX   T4CM2
        STX   T4CM2+2
T4B4A   NOP            NOP/RTS
        LDX   #T4AM1   LOAD MSG PNTR
        JSR   TESTDK   PRINT
*       TEST 5 - TRACK STEP TIME
T5      LDX   #T5S     RETRY ADDR
        STX   RETRY
        LDX   #T5MSG   STEP TIME
        JSR   PSTRNG   PRINT
        JSR   RESTOR
T5S     LDX   #$0800   20.48MSEC
        STX   STPCMP   TIM SCRATCH
        LDA   CLKFLG   IF 1MHZ
        BNE   T5B0     DIV BY 2
        LSR   STPCMP
        ROR   STPCMP+1
T5B0    LDX   STPCMP   INIT
        STX   STPSIZ   STEP TIME
        LDA   #$09     INIT LOOP CNT
        STA   TEMP
T5B1    LDA   #$20     INIT STEP CNT
        STA   CNTR
        STA   11,Y     DATA REG
        LDA   #$1B     SEEK CMD
        STA   8,Y      CMD REG
        LDX   #$5400   INIT LP CNTR
T5B2    JSR   DELAY    BUSY LOOP
        LEAX  -1,X     .
        BEQ   T5AERR   .
        LDB   8,Y      .
        BITB  #$01     .
        BNE   T5B2     .
        BRA   T5B3
T5AERR  INC   ERRFLG   ERROR
        LDX   #T5EMSG
        JSR   PSTRNG
        JSR   OPTION
        LBRA  T6       SKIP T5C
T5B3    JSR   STPRTE   VARIABLE STP
        DEC   CNTR     AT TRKD?
        BNE   T5B3
        LSR   STPCMP   TIME/2
        ROR   STPCMP+1
        JSR   DELTS
        LDA   8,Y      STATUS
        BITA  #$04     TRKD?
```

```
        BEQ   T5B4    GT TRKD
        LDD   STPSIZ  STP=STP-TIME
        SUBD  STPCMP
        STD   STPSIZ
        BRA   T5B5
T5B4    JSR   RESTOR  GOTO TRKD
        LDD   STPSIZ  STP=STP+TIME
        ADDD  STPCMP
        STD   STPSIZ
T5B5    CLR   9,Y
        DEC   TEMP    DELTA=.08MSEC?
        BNE   T5B1    LOOP
        LDX   STPSIZ
        STX   COUNT+1
        LDU   #T5MSI  MESSAGE ADDR
        JSR   CLKCMP  CLOCK COMPENS
        LDD   #$0000  LO=DSEC
        STD   LIMT
        LDD   #$0FA0  HI=40MSEC
        JSR   PASFAL  LIM OK?
        JSR   UPDMSG
        LDX   #T5MSH
        JSR   TESTOK  PRINT
*       TEST 6 - DOUBLE DENSITY STATUS (E0X4)
T6      LDX   #T6S    RETRY ADDR
        STX   RETRY
        LDX   #T6MSG  DOUB DENS STAT
        JSR   PSTRNG
T6S     CLR   TEMP
        LDA   #$0F    RESTORE
        LDX   #$0000  SET COUNTER
        STA   8,Y     CMD REG
        JSR   DELAY
T6A     BRN   T6A     DELAY
        LDB   4,Y     DD STAT
        BITB  #$40    INTRQ?
        BNE   T6A1    NOT BUSY
        LEAX  1,X     COUNT
        BNE   T6A
T6A1    CMPX  #$0000  ERROR
        BEQ   T6AF
        LDX   OK      OK
        BRA   T6A2
T6AF    INC   ERRFLG  FAIL
        INC   TEMP
        LDX   NGO
T6A2    STX   T6MS1+32
        LDX   #T6MS1  INTRQ(D6)
        JSR   PSTRNG  PRINT
        LDA   ERRFLG  ERROR?
        BEQ   T6B     NO
        JSR   OPTION
        LDA   TEMP
        BNE   T7
T6B     LDX   #$0000  DATA PNTR
        LDB   #$01    SECT 1
        STB   10,Y    SECT REG
        JSR   DELAY
        LDA   #$8C    READ SECT
        JSR   RDDAT2  DD READ
        STX   COUNT+1 NO OF BYTES
        BITB  #18     ERRORS?
        BEQ   T6BOK
        INC   ERRFLG
        LDX   #NGO
        BRA   T6BOK1
T6BOK   LDX   #OK
        INC   DDFLG   DDSTAT OK
T6BOK1  LDU   #T6MS2+34 MSG PNTR
        LDA   #$02    2 CHAR
        JSR   XFER
        LDX   #T6MS2  DRQ =
        JSR   TESTOK  PRINT
T6C     LDX   COUNT+1 NO OF BYTES
        CMPX  #$0100  256?
        BEQ   T6C1    YES
        INC   ERRFLG
        CLR   T6FLG   T6 FAILED
        BRA   T6C2
T6C1    LDA   #$01
        STA   T6FLG
T6C2    LDU   #T6MS3A #BYTES
        JSR   UPDMSG
        LDX   #$0000  CLR MSB NO
        STX   T6MS3A
        LDX   #T6MS3
        JSR   TESTOK  PRINT
*TEST 7 - RADIAL ALIGNMENT TEST
T7      LDX   #T7MSG  READ DISK
        JSR   PSTRNG PRINT
        LDD   #$3631  SET HI/LO
        JSR   CHKIN   GET CHAR
        STA   T7OPTN
        LDX   #EXTD
        STX   EXIT+1
        CMPA  #$33    EXIT?
        LBEQ  EXIT
        CMPA  #$35    NEXT TEST?
        LBEQ  T8
        LDX   #T7
        STX   EXIT+1
        BLO   T7S1
        JSR   PCRLF
        LDX   #T7SYTB
        JSR   PSTRNG
        BRA   T7
T7S1    JSR   RESTOR
        CLR   ERRFLG
        LDX   #T7FORM GET DSK FORMAT
        JSR   PSTRNG
        LDD   #$3431  SET HI/LO
        JSR   CHKIN   GET CHAR
        ANDA  #$DF    CONV TO BIN
        SUBA  #$01    REMOVE OFFSET
        STA   FORMAT
        LDX   #T7SECT GET TABLE
        LDB   A,X     GET MAX SECT
        STB   MAXSEC
        LSRA          DD BIT
        STA           DNSFLG
T7A     LDX   #T7B2   RETRY ADDR
        STX   RETRY
        LDA   T7OPTN
        CMPA  #$31    ALL TRKS
        BEQ   T7B
        CMPA  #$32    ALIGN?
        BNE   T7A1
        LDX   #T7MS2  DISK TYPE
        JSR   PSTRNG
        LDD   #$3231  SET HI/LO
        JSR   CHKIN
        CMPA  #$31    OP-SYS DISK
        LBNE  T7DA
T7A1    LDX   #T7TRK  TRK NO?
        JSR   PSTRNG  PRINT
        JSR   GETDEC
        CMPA  #$29
        BHI   T7A1
        STA   CNTR    SCRAMBLE
        JSR   SCRAMB
        LDA   TRK
        STA   CNTR
        INC   ERRFLG  STOP AFTER TRK
        LDA   T7OPTN
        CMPA  #$32    ALIGN?
        BEQ   T7C
```

To Be Continued

Classifieds

As Submitted - No Guarantees

Surplus Unused Motorola VME Modules & Electronic Solutions Enclosures for Sale at Discount

MVME133	CPU Module-68020, 1MB DRAM, 68881 FPP, 3 serial Ports, EPROM Sockets, VMEbus Interface	$675
MVME225-1	1MB DRAM Module, A32/D32 VMEbus Interface	$380
MVME320A	Winchester / Fllopy Controller	$490
MVME332	8 Channel intelligent Serial Communications Module	$675
Series 7	Electronic Solutions 7 Slot Desktop Enclosure, P1/P2 Backplane, 325W PS, Space for Winchester/Floppy/Tape	$695

Respond to: John Gannon, RPG, P.O. Box C12399, Ste 162, Scottsdale, Arizona 85267 Phone (602) 951-3373

***

CDS--1, 20 Meg Hard Disk w/controller $100
S+ Memory Cards, CPU Cards, Hard Disks w/Controller Cards, I/O Cards, Cabinets, Power Supplies.
S/09 CPU Cards, Memory, I/O Cards, Controller Cards, Cabinets, Power Supplies
3-Dual 8" drive enclosure with power supply. New in box. $125 each.
5-Siemens 8" Disk Drive, $100 each.

Tom (615) 842-4600 M-F 9AM to 5PM EST

***

Motorola VME-10 with hi resolution monochrome monitor, hard disk, serial and parallel cards, Pascal, assembler, linker and documentation. Almost new $4500.
One SSB-6809 system with hard disk, miscellaneous software. Make Offer
Cadwell Laboratories, 909 N. Kellogg Street, Kennweick, WA 99336 (509) 735-6481

FLEX™/SK-DOS™/MS-DOS™ Transfer Utilities

For 68020 and CoCo* OS-9 Systems
Now READS
WRITE -DIR - DUMP - EXPLORE
FLEX, SK-DOS & MS-DOS Disk

These Utilities come with a rich set of options allowing the transfer of text type files from/to FLEX & MS-DOS disks.
*CoCo systems require the D.P. Johnson SDISK utilities and OS-9 and two drives of which one must be a "host" floppy.

CoCo Version: $69.95 68020 Version $99.95

S.E. Media
PO Box 849
5900 Cassandra Smith Rd.
Hixson, TN 37343
(615) 842-7990
FAX (615) 842-4600

MASTERCARD VISA

SPECIAL - ATARI™ & OS-9™

NOW! - If you have either the Atari 520 or 1040 - you can take advantage of the *"bargain of a lifetime"* OS-9 68K and BASIC all for the low, low price of:

$150.00

Call or Write
S.E. Media
5900 Cassandra Smith Rd.
Hixson, TN 37343
615 842-4601
FAX (615)842-7990

SK*DOS®/68K

Read the fine print to see what's in SK*DOS/68K:

☐ Full DOS documentation plus on-line help ☐ Multiple directories ☐ User-installable device drivers ☐ Install up to 8 different I/O devices ☐ Keyboard type-ahead ☐ Print-screen ☐ Virtual (RAM) disk ☐ Disk cache ☐ Up to 10 drives ☐ 5¼" or 3½" floppy drives ☐ Hard drives to 64 megabytes each ☐ I/O redirection to drives or I/O ☐ Time/date stamping of files ☐ File or disk write protect (even hard disk) ☐ Batch files ☐ Support for 68000, 68010, 68020 ☐ Monochrome or color video board support ☐ Read and write MS-DOS disk files ☐ 6809 Emulator ☐ Powerful utilities such as copy-by-date, undelete, show differences between files, prompted delete, text file browse, and more - all included ☐ Simple Basic included ☐ Fast assembler included ☐ Line editor included ☐ User support via newsletter and BBS ☐ Available software: C compiler, full Basic, screen editors, disassemblers, cross-assemblers, spelling checker, text formatter, music editor, hard disk manager, ROM-based debugger, modem communications programs, etc. More compilers coming. ☐ (Some features may not be implemented in all hardware manufacturers' implementations.)

Individual copies of SK*DOS/68K are $140; less in quantity or when bundled with hardware. Send for our 6809 / 68K hardware and software catalog. Also available as part of our hardware/software educational course.

STAR-K Software Systems Corp.
P. O. Box 209J
Mt. Kisco, NY 10549
(914) 241-0287
BBS (914) 241-3307 ◊ Fax (914) 241-8607

THE QUICK ED
OS-9 SCREEN EDITOR

Word Processor and Text Formatting System

* Edits up to nine files simultaneously
* Microjustifies mixed proportional text
* Uses **all** the fonts *on* your **printer** ½£ë°æµ
* Uses cursor and function keys *on any terminal*
* Definable high-level formatting commands
* User-specific keyboard mapping and macros
* Highly configurable on multi-user systems
* Automatic hyphenation and table of contents
* Imports graphics and listings into documents
* 450 pages of comprehensive documentation

At last ! Use your laser printer !

The **Quick Ed** *screen editor and formatter: $275.00*

Call for information on OS-9 consultancy services, disk caching, graphics, lint, Modula-2 and file managers.
'OS-9' is a Trademark of Microware Inc.

Windsor Systems
2407 Lime Kiln Lane, Louisville, KY 40222 U.S.A
Telephone: (502) 425 9580 Fax: (502) 426 3944

SOFTWARE

68000 C CROSS-COMPILER
$100 - SKDOS,MSDOS,UNIX,XENIX (OBJECT ONLY)

Aumpus K&R C language, generates 68000 assembler code
includes 68010 cross-assembler, libraries provided for SKDOS, but may be modified

CROSS-ASSEMBLERS WITH MACRO CAPABILITIES
EACH $50-FLEX,OS9,UNIFLEX,MSDOS,UNIX,SKDOS,XENIX 3/$100 ALL/$200

Specify: 180x, 6502, 6801/11, 6804, 6805, 6809, Z8, Z80, 8048, 8051, 8085, 68010, 32000
Modular cross-assemblers in C, with load/unload utilities. Sources for additional $50 each, $100 for 3, $300 for all

CMODEM TELECOMMUNICATIONS PROGRAM
$100-MSDOS,SKDOS,UNIX,FLEX-OS9,XENIX,UNIFLEX OBJECT-ONLY: EACH $50

Menu-driven with terminal mode, file transfer, MODEM7, XON/XOFF, etc.

SUPER SLEUTH DISASSEMBLERS
EACH $99-FLEX $101-OS9 $100,UNIFLEX OBJECT-ONLY: EACH $50 FLEX,OS9,COCO

Interactively generate source on disk with labels, includes xref, binary editing
Specify 6800,1,2,3,5,8,9/6502 version or Z80/8080,5 version
COCO DOS available in 6800,1,2,3,5,8,9/6502 version (not Z80/8080,5) only
68010 version $100-FLEX,OS9,UNIFLEX,MSDOS,UNIX,SKDOS,XENIX

DEBUGGING SIMULATORS FOR POPULAR 8-BIT CPUS
EACH $75-FLEX $100-OS9 $80-UNIFLEX OBJECT-ONLY: EACH $50-COCO FLEX, COCO OS9

Interactively simulate processor, includes disassembly formatting, binary editing
Specify for 6800/1, (14)6805, 6502, 6809 OS9 only, Z80 FLEX only

ASSEMBLER CODE TRANSLATORS FOR 6502, 6800/1, 6809
6502 to 6809 $75-FLEX $85-OS9 $80-UNIFLEX
6809/1 to 6809 & 6809 to portable $50-FLEX $75-OS9 Only $60-UNIFLEX

FULL-SCREEN XBASIC PROGRAMS with cursor control AVAILABLE FOR FLEX, UNIFLEX, AND MSDOS

Display Generator / Documentor	$50 w/source,	$25 without
Mailing List System	$100 w/source,	$50 without
Inventory with MRP	$100 w/source,	$50 without
Tabula Rasa Spreadsheet	$100 w/source,	$50 without

DISK AND XBASIC UTILITY PROGRAM LIBRARY
$50-FLEX $30-UNIFLEX/MSDOS

Edit disk sectors, sort directory, maintain master catalog, do disk scrub, resequence some or all BASIC program, xref BASIC program, etc. non-FLEX versions include sort and resequencer only

PROFESSIONAL SERVICES
FOR THE COMPUTING COMMUNITY

CUSTOMIZED PROGRAMMING

We will customize any of the programs described in this advertisement or in our brochure for specialized customer use or to cover new processors; the charge for such customization depends upon the marketability of the modifications.

CONTRACT PROGRAMMING

We will create new programs or modify existing programs on a contract basis, a service we have provided for over twenty years, the computers on which we have performed contract programming include most popular models of mainframes, including IBM, Burroughs, Univac, Honeywell, most popular models of minicomputers including DEC, IBM, DG, HP, AT&T, and most popular brands of microcomputers, including 6800/1, 6809, Z80, 6502, 680x0, using most appropriate languages and operating systems, on systems ranging in size from large telecommunications to single board controllers; the charge for contract programming is usually by the hour or by the task.

CONSULTING

We offer a wide range of business and technical consulting services, including seminars, advice, training, and design, on any topic related to computers; the charge for consulting is normally based upon time, travel, and expenses.

Computer Systems Consultants Inc.
1454 Latta Lane
Conyers, Georgia 30207
(404) 483-4570 • (404) 483-1717

Contact us about catalog, dealer, discounts, and services. Most programs in source; give computer, OS, disk size. 25% off multiple purchases of same program on one order. VISA and MASTER CARD accepted. Add GA sales tax (if in GA) and 5% shipping. (UNIFLEX tm Technical Systems Consultants; OS/9 Microware COCO Tandy;MSDOS Microsoft; SKDOS Stark Software

K-BASIC™

The Only 6809 BASIC to Binary Compiler for OS-9
FLEX or SK*DOS
Even runs on the 68XXX SK*DOS Systems*

Hundreds Sold at Suggested Retail:

~~$199.00~~

For a LIMITED time save over 65%... This sale will not be repeated after it's over! *

SALE SPECIAL:

$69.95

SPECIAL

Thank-You-Sale

Only From:

CPI **S.E. Media™**

5900 Cassandra Smith Rd.
Hixson, Tn 37343
Telephone 615 842-6809

- 6809 - OS-9™ users can now transfer their FLEX™ Extended BASIC (XBASIC) source files to OS-9, compile with the OS-9 version and run them as any other OS-9 binary "CMD" program. Much faster than BASIC programs.

- 6809 - FLEX users can compiler their BASIC source files to a regular FLEX ".CMD" file. Much faster execution.

- 68XXX - SK*DOS™ users running on 68XXX systems (such as the Mustang-08/A) can continue to execute their 6809 FLEX BASIC and compiled programs while getting things ported over to the 68XXX. SK*DOS allows 6809 programs to run in emulation mode. This is the only system we know of that will run both 6809 & 68XXX binary files.

K-BASIC is a true compiler. Compiling BASIC 6809 programs to binary command type programs. The savings in RAM needed and the increased speed of binary execution makes this a must for the serious user. And the price is now RIGHT!

Don't get caught up in the *"Learn a New Language"* syndrome - Write Your Program In BASIC, Debug it in BASIC and Then Compile it to a .CMD Binary File.

A Division of Computer Publishing Inc.
Over 1,200 Titles - 6800-6809-68000
FAX (615)842-7990

* K-BASIC will run under 68XXX SK*DOS in emulation mode for the 6809.

Price subject to change without notice.

THE 6800-6809 BOOKS

OS - 9 User Notes

By: Peter Dibble

The publishers of 68' Micro Journal are proud to make available the publication of Peter Dibbles

OS9 USER NOTES

Information for the BEGINNER to the PRO, Regular or CoCo OS9

Using OS9

HELP, HINTS, PROBLEMS, REVIEWS, SUGGESTIONS, COMPLAINTS, OS9 STANDARDS, Generating a New Bootstrap, Building anew System Disk, OS9 Users Group, etc.

Programming Languages

Assembly Language Programs and Interfacing; Basic09, C, Pascal, and Cobol reviews, programs, and uses; etc.

Disks Include

No typing all the Source Listings in. Source Code and, where applicable, assembled or compiled Operating Programs. The Source and the Discussions in the Columns can be used "as is", or as a "Starting Point for developing your OWN more powerful Programs. Programs sometimes use multiple Languages such as a short Assembly Language Routine for reading a Directory, which is then "piped" to a Basic09 Routine for output formatting, etc.

BOOK $9.95

Typeset — w/ Source Listings
(3-Hole Punched; 8 x 11)
Deluxe Binder $5.50

All Source Listings on Disk

1-8" SS, SD Disk **$14.95**
2-5" SS, SD Disks **$24.95**

FLEX USER NOTES

By: Ronald Anderson

The publishers of 68 MICRO JOURNAL are proud to make available the publication of Ron Anderson's **FLEX USER NOTES**, in book form. This popular monthly column has been a regular feature in 68' MICRO JOURNAL SINCE 1979. It has earned the respect of thousands of 68 MICRO JOURNAL readers over the years. In fact, Ron's column has been described as the 'Bible' for 68XX users, by some of the world's leading microprocessor professionals. The most needed and popular 68XX book available. Over the years Ron's column has been one of the most popular in 68 MICRO JOURNAL. And of course 68 MICRO JOURNAL is the most popular 68XX magazine published.

Listed below are a few of the **TEXT** files included in the book and on diskette. All TEXT files in the book are on the disks.

LOGO.C1	File load program to offset memory - ASM PIC
MEMOVES.C1	Memory move program - ASM PIC
DUMP.C1	Printer dump program - uses LOGO - ASM PIC
SUBTEST.C1	Simulation of 6800 code to 6809, show differences - ASM
TERMEM.C2	Modem input to disk (or other port input to disk) - ASM
M.C2	Output a file to modem (or another port) - ASM
PRINT.C3	Parallel (enhanced) printer driver - ASM
MODEM.C2	TTL output to CRT and modem (or other port) - ASM
SCIPKG.C1	Scientific math routines - PASCAL
U.C4	Mini-monitor, disk resident, many useful functions - ASM
PRINT.C4	Parallel printer driver, without PFLAG - ASM
SET.C5	Set printer modes - ASM
SETBAS1.C5	Set printer modes - A-BASIC

Note: .C1, .C2, etc.=Chapter 1, Chapter 2, etc.

** Over 30 TEXT files included is ASM (assembler)-PASCAL-PIC (position independent code) TSC BASIC-C, etc.

Book only: **$7.95** + $2.50 S/H

With disk: 5" **$20.90** + $2.50 S/H
With disk: 8" **$22.90** + $2.50 S/H

Shipping & Handling $3.50 per Book, $2.50 per Disk set
Foreign Orders Add $4.50 Surface Mail
or &7.00 Air Mail
If paying by check - Please allow 4-6 weeks delivery
*** All Currency in U.S. Dollars**
Continually Updated in 68 Micro Journal Monthly

Computer Publishing Inc.
5900 Cassandra Smith Rd.
Hixson, TN 37343
Telephone (615) 842-4601
FAX (615) 842-7990

MASTER CARD

VISA

FLEX is a trademark of Technical Systems Consultants
OS9 is a trademark of Microware and Motorola Telex 5106006630
68' Micro Journal is a trademark of Computer Publishing Inc.

68' Micro Journal
Reader Service Disks

Disk- 1	Filesort, Miniicat, Minicopy,Minifms, **Lifetime, **Poetry, **Foodlist, **Diet.
Disk- 2	Diskedit w/ inst.& fixes. Prime, *Prmod, **Snoopy, **Foothall, **Hexpawn,**Lifetime.
Disk- 3	Cbug09, Sec1, Sec2, Find, Table2, Intext, Disk-exp, *Disksave.
Disk- 4	Mailing Program, *Finddat, *Change, *Testdisk.
Disk- 5	*DISKFIX 1, *DISKFIX 2, **LETTER, **LOVESIGN, **BLACKJAK, **BOWLING.
Disk- 6	**Purchase Order, Index (Disk file indx).
Disk- 7	Linking Loader, Rload, Hatkness.
Disk- 8	Crest, Lzspher (May 82).
Disk- 9	Datecopy, Diskfix9 (Aug 82).
Disk-10	Home Accounting (July 82).
Disk-11	Dissembler (June 84).
Disk-12	Modem68 (May 84).
Disk-13	*Initmf68, Testmf68,*Cleanup,*Dskalign,Help,Date.Txt.
Disk-14	*Init, *Test, *Terminal, *Find, *Diskedit, Init.Lib.
Disk-15	Modem9 + Updates (Dec. 84 Gilchrist) to Modem9 (April 84 Commo).
Disk-16	Copy.Txt, Copy.Doc, Cat.Txt, Cat.Doc.
Disk-17	Match Utility, RATBAS, A Basic Preprocessor.
Disk-18	Parse.Mod, Size.Cmd (Sept. 85 Armstrong),CMDCODE, CMD.Txt (Sept. 85 Spray).
Disk-19	Clock, Date, Copy, Cat, PDEL.Asm & Doc., Errors.Sys, Do, Log.Asm & Doc.
Disk-20	UNIX Like Tools (July & Sept. 85 Taylor & Gilchrist). Dragon.C, Grep.C, LS.C, FDUMP.C.
Disk-21	Utilities & Games - Date, Life, Madness, Touch, Goblin, Starshot, & 15 more.
Disk-22	Read CPM & Non-FLEX Disks. Fraser May 1984.
Disk-23	ISAM, Indexed Sequential file Accessing Methods, Condon Nov.85. Extensible Table Driven. Language Recognition Utility, Anderson Mar86.
Disk-24	68' Micro Journal Index of Articles & Bit Bucket Items from 1979 - 1985, John Current.
Disk-25	KERMIT for FLEX derived from the UNIX ver. Burg Feb. 1986. (2)-5" Disks or (1)-8" Disk.
Disk-26	Compacta UniBoard review, code & diagram, Burlison March '86.
Disk-27	ROTABIT.TXT, SUMSTEST.TXT, CONDATA.TXT, BADMEN.TXT.
Disk-28	CT-82 Emulator, bit mapped.
Disk-29	**Star Trek
Disk-30	Simple Winchester, Dec.'86 Green.
Disk-31	*** Read/Write MS/PC-DOS (SK*DOS)
Disk-32	Heir-UNIX Type upgrade - Feb .87
Disk-33	Build the GT-4 Terminal - Nov. 87 Joseph Condon.
Disk-34	FLEX 6809 Diagnostics, Disk Drive Test, ROM Test, RAM Test - Apr. 89Koipi.
Disk-35	DO A FLEX-09 Batch File Processor - Oct. 88 - Dave Howland
Disk-36	Add Graphics To Your SBC - Nov. 88 - Joseph Condon
Disk-37	Minix for the PT68K-2 - June/July 89 - J. GaryMills

NOTE:

This is a reader service ONLY! No Warranty is offered or implied, they are as received by 68' Micro Journal, and are for reader convenience ONLY (some MAY include fixes or patches). Also 6800 and 6809 programs are mixed, as each is fairly simple (mostly) to convert to the other. Software is available to cross-assemble all.

* Denotes 6800 - ** Denotes BASIC

*** Denotes 68000 - 6809 no indicator.

8" disk $19.50

5" disk $16.95

Shipping & Handling -U.S.A. Add: - $3.50

Overseas add: $4.50 Surface - $7.00 Airmail

68' MICRO JOURNAL

5900 Cassandra Smith Rd.

Hixson, TN 37343

(615) 842-4600

FAX (615) 842-7990

!!! Subscribe Now !!!
68' MICRO JOURNAL

OK, PLEASE ENTER MY SUBSCRIPTION

Bill My: ☐ Mastercard ☐ VISA

Card #_____ Exp. Date_____

For 1 Year _____ 2 Years _____ 3 Years _____

Enclosed: $_____

Name _____

Street _____

City _____ **State** _____ **Zip** _____

Country _____

My Computer is: _____

My Operating System is: _____

Subscription Rates

U.S.A.: 1 Year $24.50, 2 Years $42.50, 3 Years $64.50

***Foreign Surface: Add $12.00 per Year to USA Price.**

***Foreign Airmail: Add $48.00 per Year to USA Price.**

***Canada & Mexico: Add $9.50 per Year to USA Price.**

***U.S. Currency Cash or Check Drawn on a USA Bank !**

68' Micro Journal

5900 Cassandra Smith Rd.
POB 849
Hixson, TN 37343
Telephone 615 842-4600
FAX (615) 842-7990

VISA

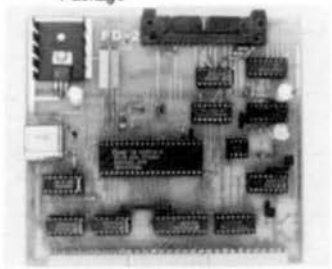
master card

IF YOU NEED IT, WE'VE GOT IT!
(OR WE'LL MAKE IT)

Yes, **PERIPHERAL TECHNOLOGY** still sells the FD-2 for SS-50 Bus Computers. And, if you don't need it, we sell other products from Single Board Computers to Systems which should fit your requirements. Customs Hardware Design is also available. Here's a small sample of what we offer:

FD-2 FLOPPY DISK CONTROLLER

- Controls up to four 5 1/4" Drives
- Runs in 1 or 2 MHZ Systems
- Can be configured for either 4 or 16
- Addresses per I/O Slot (SS30 or SS30C)
- Uses WD2797 Controller Chip (compatible with 1771/179X Controller Chip)
- Hardware and Software compatable with SWTPC DC-4 controllers
- 6800/6809 Flex Drivers available
- SK*DOS Operating System or OS9/6809 Driver Package



OS9 is a Trademark of Microware & Motorola
SWTPC is a Trademark of Southwest Technical Systems

PT69-4 SINGLE BOARD COMPUTER

- 6809E Processor/1 MHZ Clock
- Four RS232 Serial Ports using 6850'S
- Two 8-Bit parallel Ports using 6821 PIA
- Time-Of-Day Clock (MC146818)
- 59K of user RAM
- 2K or 4K of EPROM using 2716 or 2732
- Double Sided/Double Density Floppy Controller
- Can Read/Write Radio Shack OS/9 Diskettes
- Board Size 5.6" x 8.2"

MONTHLY SPECIAL
FD-2 Floppy Disk Controller: **$129.00**
Regular Price: $149.00

SYSTEMS

- Floppy or Winchester Versions available
- Systems use any 6809 Single Board Computer or 68008 Board
- Will be configured to meet your Requirements
- OS9 & SK*DOS Operating Systems
- Call or write for system configurations

PERIPHERAL TECHNOLOGY
1710 Cumberland Point Dr. Suite 8
Marietta, Georgia 30067
(404) 984-0742 Telex# 880584
VISA/ MASTERCARD/ CHECK/ C.O.D.

Catalogues Available Upon Request.

DATA-COMP SPECIAL

Heavy Duty Power Supplies

For A limited time our HEAVY DUTY SWITCHING POWER SUPPLY. These are BRAND NEW units. Note that these prices are less than 1/4 the normal price for these high quality units.

Make: Boschert
Size: 10.5 x 5 x 2.5 inches
Including heavy mounting bracket and heatsink.

Rating: in 110/220 volts ac (strap change) Out: 130 watts

Output: +5v - 10 amps
+12v - 4.0 amps
+12v - 2.0 amps
-12v - 0.5 amps

Mating Connector: Terminal strip
Load Reaction: Automatic short circuit recovery
SPECIAL: $59.95 each
2 or more $49.95 each
Add: $7.50 each S/H

Make: Boschert
Size: 10.75 x 6.2 x 2.25 inches

Rating: 110/220 ac (strap change) Out: 81 watts

Outputs: +5v - 8.0 amps
+12v - 2.4 amps
+12v - 2.4 amps
+12v - 2.1 amps
-12v - 0.4 amps

Mating Connectors: Molex
Load Reaction: Automatic short circuit recovery
SPECIAL: $49.95 each
2 or more $39.95 each
Add: $7.50 S/H each

5900 Cassandra Smith Rd., Hixson, Tn. 37343 Telephone 615 842-4600 Telex 510 600-6630 Fax (615)842-7990

An Ace of a System in Spades! The New

MUSTANG-08/A™

Now with 4 serial ports standard & speed increase to 12 Mhz CPU + on board battery backup and includes the PROFESSIONAL OS-9 package - including the $500.00 OS-9 C compiler! This offer won't last forever!

NOT 128K, *NOT* 512K

FULL 768K No Wait RAM

The MUSTANG-08™ system took every hand from all other 68008 systems we tested, running OS-9 68K!

The MUSTANG-08 includes OS9-68K™ and/or Peter Stark's SK*DOS™. SK*DOS is a single user, single tasking system that takes up where *FLEX™ left off. SK*DOS is actually a 68XXX FLEX type system *(Not a TSC product.)*

The OS-9 68K system is a full blown multi-user, multi-tasking 68XXX system. All the popular 68000 OS-9 software runs. It is a speed *whiz on disk I/O.* Fact is, the MUSTANG-08 is faster on disk access than some other 68XXX systems are on memory cache access. Now, that is fast! And that is just a small part of the story! See benchmarks!

SALE!!!

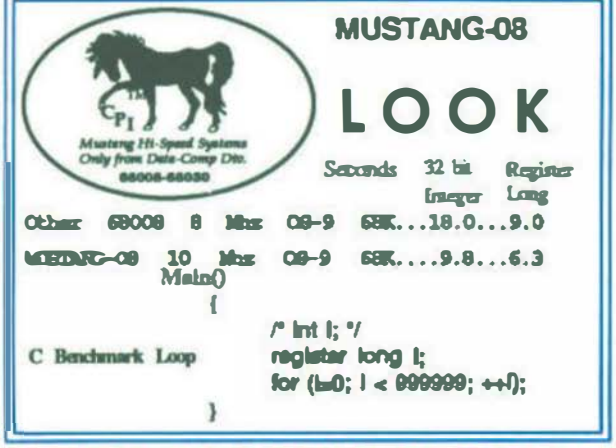
System includes OS-9 68K or SK*DOS - Your Choice

Specifications:

CPU	MC68008	12 Mhz
RAM	768K No Wait States	256K Chips
PORTS	4 - RS232 2 - 8 bit Parallel	MC68681 DUART MC6821 PIA
CLOCK	MK48T02	Real Time Clock Bat. B/U
EPROM	16K, 32K or 64K	Selectable
FLOPPY	WD1772	5 1/4 Drives
HARD DISK	Interface Port	WD1002 Board

Now more serial ports - faster CPU Battery B/U - and $850.00 OS-9 Professional with C compiler included!

***$400.00** See Mustang-02 Ad - page 5 for trade-in details



Now even faster! with 12 Mhz CPU

C Compile times: OS-9 68K Hard Disk	
MUSTANG-08 8 Mhz CPU	0 min - 32 sec
Other popular 68008 system	1 min - 05 sec
MUSTANG-020	0 min - 21 sec

25 Megabyte Hard Disk System

$2,398.90

Complete with PROFESSIONAL OS-9 includes the $500.00 C compiler, PC style cabinet, heavy duty power supply, 5" DDDS 80 track floppy, 25 MegByte Hard Disk - Ready to Run

Unlike other 68008 systems there are several significant differences. The MUSTANG-08 is a full 12 Megahertz system. The RAM uses NO wait states, this means full bore MUSTANG type performance.

Also, allowing for addressable RDM/PROM the RAM is the maximum allowed for a 68008. The 68008 can only address a total of 1 Megabytes of RAM. The design allows all the RAM space (for all practical purposes) to be utilized. What is not available to the user is required and reserved for the system.

A RAM disk of 480K can be easily configured, leaving 288K free for program/system RAM space. The RAM DISK can be configured to any size your application requires (system must have 128K in addition to its other requirements). Leaving the remainder of the original 768K for program use. Sufficient source included (drivers, etc.)

FLEX is a trademark of TSC MUSTANG-08 is a trademark of CPI

Data-Comp Division

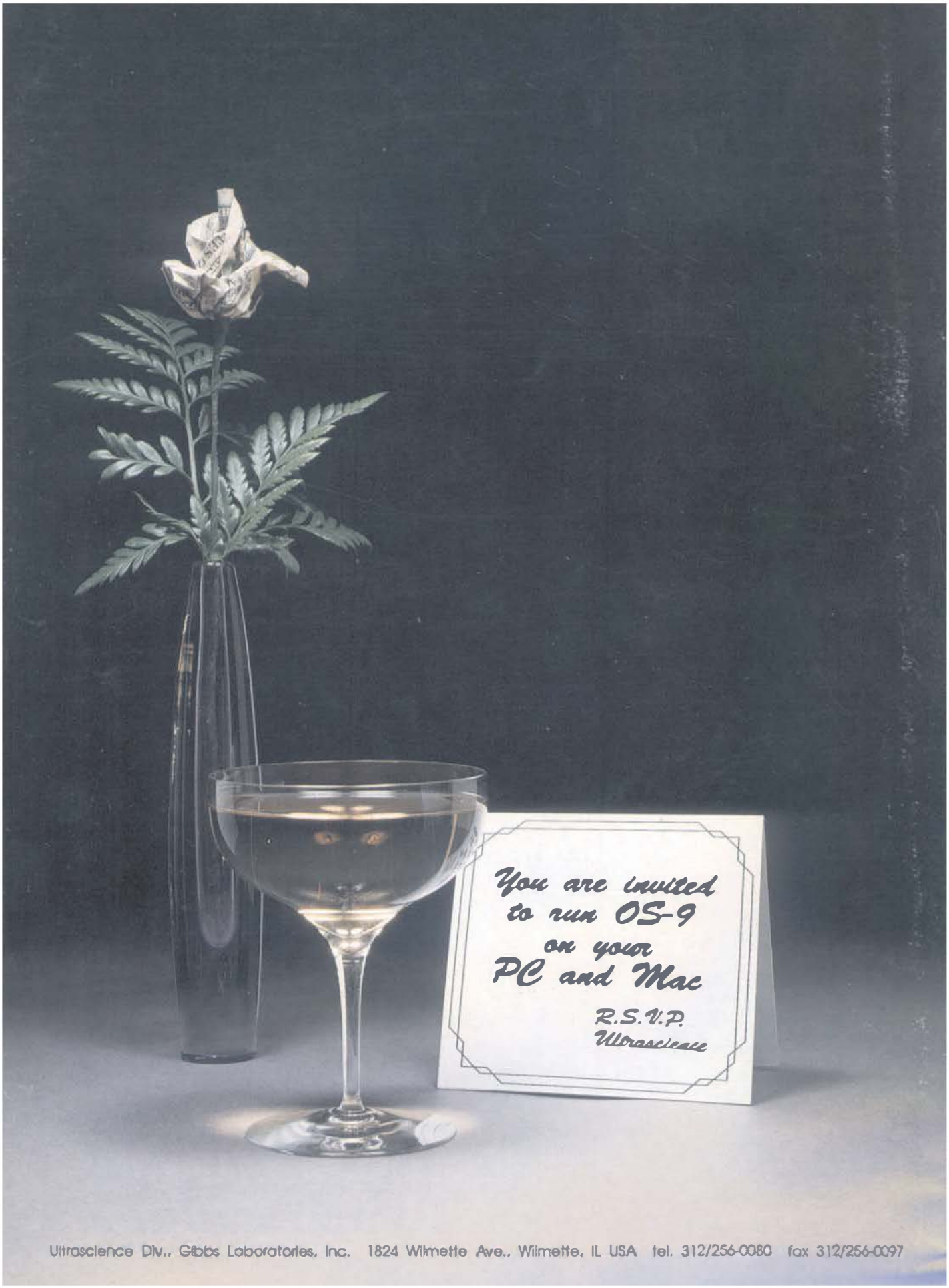
A Decade of Quality Service"

Systems World-Wide

Computer Publishing, Inc. 5900 Cassandra Smith Road

Telephone 615 842-4601 - Telex 510 600-6630 Hixson, Tn 37343

* Those with SWTPC Hi-density FLEX 5" - Call for special info.



Ultrascience Div., Gibbs Laboratories, Inc. 1824 Wilmette Ave., Wilmette, IL USA tel. 312/256-0080 fax 312/256-0097